Discrete-time differentiators: design and comparative analysis

Mohammad Rasool Mojallizadeh, Bernard Brogliato, Vincent Acary

Discrete-time differentiators: design and comparative analysis

Mohammad Rasool Mojallizadeh[*1], Bernard Brogliato[†1], and Vincent Acary[‡1]

[1]Univ. Grenoble Alpes, INRIA, CNRS, Grenoble INP, LJK, 38000 Grenoble, France

February 5, 2021

Abstract

This work deals with the problem of online differentiation of noisy signals. In this context, several types of differentiators including linear, sliding-mode based, adaptive, Kalman, and ALIEN differentiators are studied through mathematical analysis and numerical experiments. To resolve the drawbacks of the exact differentiators, new implicit and semi-implicit discretization schemes are proposed in this work to suppress the digital chattering caused by the wrong time-discretization of set-valued functions as well as providing some useful properties, e.g., finite-time convergence, invariant sliding-surface, exactness. A complete comparative analysis is presented in the manuscript to investigate the behavior of the discrete-time differentiators in the presence of several types of noises, including white noise, sinusoidal noise, and bell-shaped noise. Many details such as quantization effect and realistic sampling times are taken into account to provide useful information based on practical conditions. Many comments are provided to help the engineers to tune the parameters of the differentiators.

Contents

1 Introduction **10**

2 Continuous-time differentiators **12**

2.1 Slotine-Hedrick-Misawa set-valued differentiator 13

2.2 Super-twisting differentiator 13

2.3 Arbitrary-order super-twisting differentiator 14

2.4 Uniform robust exact differentiator 14

2.5 Quadratic sliding-mode differentiator 15

2.6 Homogeneous differentiator 15

[*]mohammad-rasool.mojallizadeh@inria.fr

[†]bernard.brogliato@inria.fr

[‡]vincent.acary@inria.fr

2.7 Adaptive differentiators . . . 15
2.7.1 Adaptation mechanism for the coefficients . . . 16
2.7.2 Adaptation mechanism for the exponents . . . 16
2.8 ALIEN differentiator . . . 18
2.9 High-gain differentiator . . . 19
2.10 Differentiators in closed-loop systems . . . 19

3 Discrete-time exact differentiators 20
3.1 Explicit discretization of the exact differentiators . . . 20
3.2 Explicit HD . . . 23
3.3 Implicit discretization of the exact differentiators . . . 24
3.3.1 Implicit discretization of the super-twisting differentiator . . . 24
3.3.2 Implicit discretization of the quadratic differentiator . . . 27
3.3.3 Implicit discretization of the uniform robust exact differentiator . . . 29
3.3.4 Implicit discretization of the arbitrary-order super-twisting differentiator . . . 32
3.3.5 Implicit discretization of HDD . . . 39
3.3.6 Implicit discretization of GHDD . . . 41
3.3.7 Implicit discretization of VGED . . . 42
3.3.8 Implicit discretization of the SHMD . . . 43
3.4 Semi-implicit discretization of the exact differentiators . . . 45
3.4.1 The first semi-implicit scheme . . . 46
3.4.2 The second semi-implicit scheme . . . 46
3.4.3 Semi-implicit differentiators in the literature . . . 52
3.5 Kalman differentiator . . . 53

4 Tuning the parameters of the differentiators 54
4.1 Objective functions . . . 54
4.2 Problem formulation . . . 55

5 Open-loop analysis of the differentiators 58
5.1 Performances in noise-free conditions . . . 60
5.2 Analysis of robustness against the noise . . . 64
5.2.1 Robustness against white noise . . . 64
5.2.2 Robustness against sinusoidal noise . . . 72
5.2.3 Robustness against bell-shaped noise . . . 85
5.2.4 Performance of the differentiators in the presence of quantization . . . 91
5.3 Effect of the sampling period on the differentiators . . . 95

5.4 Effect of the frequency of the input signal on the differentiators . . . 102
5.5 Behavior of the differentiators in the presence of initial error . . . 102
5.6 Higher-order differentiation . . . 110
5.7 Sensitivity of the exact differentiators to the parameters . . . 116
5.8 Sensitivity of the implicit method to the accuracy of the solver . . . 119
5.8.1 Newton's method . . . 120
5.8.2 Householder's method . . . 120
5.8.3 Bairstow's method . . . 121
5.8.4 MATLAB solver . . . 121
5.8.5 Numerical simulations with different solvers . . . 121

6 General conclusions **123**
6.1 Future works . . . 126

A Auxiliary technical result **127**

B Proofs **127**

C Effect of the criteria on the parameter tuning **131**

D Analysis of the differentiators based on the homogeneity properties **137**

E Differentiation toolbox **138**

F Quantization error **140**

List of Figures

1 A typical application of a differentiator in a control loop. . . . 10
2 Graphical interpretation of (44): $\xi_k \in \mathrm{sgn}(x_k) \Leftrightarrow x_k \in \mathcal{N}_{[-1,1]}(\xi_k)$. . . . 26
3 Flowchart of the I-STD. The block Z^{-1} indicates one-step delay. . . . 27
4 Phase-portrait of the I-QD. Input signal is $\sin(t)$. $F = 100, \alpha = 5, h = 1\text{ms}$. . . . 29
5 Flowchart of the I-URED. The block Z^{-1} indicates one-step delay. . . . 31
6 Phase-portrait of the I-URED. Input signal is $\sin(t)$. $\lambda_1 = 2, \lambda_2 = 2, h = 1\text{ms}$. . . . 32
7 Flowchart of the I-AO-STD. The block Z^{-1} indicates one-step delay. . . . 34
8 Investigation of the (95) for I-STD and E-STD. $L = 20$, $\lambda_1 = 1.1, n = 1, i = 1$ 38
9 Flowchart of the third-order I-HDD. The block Z^{-1} indicates one-step delay. . . . 40
10 Flowchart of the third-order I-GHDD. The block Z^{-1} indicates one-step delay. . . . 42

11 Output of the SHMD (5) discretized using explicit method for the input $f(t) = \sin(t)$ ($n = 1, h = 1$ms). 43
12 Flowchart of the implicit SHMD . The block Z^{-1} indicates one-step delay. 45
13 Flowchart of the second semi-implicit scheme for the STD in (125). The block Z^{-1} indicates one-step delay. 48
14 Flowchart of the second semi-implicit AO-STD (124). The block Z^{-1} indicates one-step delay. 50
15 Graphical interpretation of I-STD ($n = 2$) (45), I-URED ($n = 4$) (63), I-AO-STD ($n = 1, 2, \dots$) (76), I-HDD (n=1,2,...) (98), I-GHDD (n=1,2,...) (108), I-FDFF and I-AO-FDFF ($n = 1$) (112), SI-STD (n=1) (127), and SI-AO-STD (n=1) (132). Note that for the Case 2, the plots are only provided for $b_k = 0$. 51
16 First-order differentiation under a noise-free condition. Parameters are shown in Table 3 ($h = 50$ms). 61
17 Error of the first-order differentiation under a noise-free condition. Parameters are shown in Table 3 ($h = 50$ms). 61
18 First-order differentiation in the presence of white noise. Parameters are shown in Table 3 ($h = 50$ms, SNR=30dB) 65
19 Error of the first-order differentiation in the presence of white noise. Parameters are shown in Table 3 ($h = 50$ms, SNR=30dB). 66
20 $\bar{L}_2$ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms). 68
21 $\tilde{L}_2$ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms). 69
22 L_∞ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms). 70
23 Variation for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms). 71
24 THD for the first-order differentiation concerning SNR. Parameters are shown in Table 3 ($h = 50$ms). 72
25 First-order differentiation under sinusoidal noise 0.05sin(20t). Parameters are shown in Table 6 ($h = 50$ms). 74
26 Error of the first-order differentiation under sinusoidal noise 0.05sin(20t). Parameters are shown in Table 6 ($h = 50$ms). 75
27 $\bar{L}_2$ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms). 77
28 $\tilde{L}_2$ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms). 78

29 L_∞ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms). 79
30 Variation for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms). 79
31 THD for the first-order differentiation concerning noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms). 80
32 $\bar{L}_2$ for the first-order differentiation with respect to noise amplitude. Input noise: $A\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms). 81
33 $\tilde{L}_2$ for the first-order differentiation with respect to noise amplitude. Input noise: $A\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms). 82
34 L_∞ for the first-order differentiation with respect to noise amplitude. Input noise: $A\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms). 83
35 Variation for the first-order differentiation with respect to noise amplitude. Input noise: $A\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms). 84
36 THD for the first-order differentiation with respect to noise amplitude. Input noise: $A\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms). 85
37 Spectrum analysis of different types of noises. (A, B): sinusoidal noise, (C, D): white noise, and (E, F): bell-shaped noise. 86
38 First-order differentiation under a bell-shaped noise ($h = 50$ms). 88
39 First-order differentiation error under a bell-shaped noise ($h = 50$ms). 89
40 First-order differentiation under quantization with resolution 0.1 ($h = 50$ms). 93
41 Error of the first-order differentiation under quantization with resolution 0.1 ($h = 50$ms). . . 94
42 $\bar{L}_2$ for the first-order differentiation with different sampling periods (SNR=∞). 98
43 $\tilde{L}_2$ for the first-order differentiation with different sampling periods (SNR=∞). 99
44 L_∞ for the first-order differentiation with different sampling periods (SNR=∞). 100
45 Variation for the first-order differentiation with different sampling periods (SNR=∞). 101
46 THD for the first-order differentiation with different sampling periods (SNR=∞). 102
47 Output of the differentiators in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms, SNR=30dB). 105
48 Internal variables of the VGED. Parameters are shown in Table 13 ($h = 50$ms, SNR=30dB). . 105
49 $\bar{L}_2$ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms). 106
50 $\tilde{L}_2$ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms). 107
51 L_∞ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms). 108

52 Variation for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms). 109
53 THD for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms). 110
54 Flowchart of the third-order cascade configuration of differentiators 110
55 Third-order differentiation in the presence of white noise. Parameters are shown in Table 14 ($h = 10$ms, SNR=90dB). 113
56 $\bar{L}_2$ for the third-order differentiation under different SNRs (h =10ms). 113
57 $\tilde{L}_2$ for the third-order differentiation under different SNRs (h =10ms). 114
58 L_∞ for the third-order differentiation under different SNRs (h =10ms). 114
59 Variation for the third-order differentiation under different SNRs (h =10ms). 115
60 THD for the third-order differentiation under different SNRs (h =10ms). 116
61 output of the differentiators with an oversized differentiation gains $L = F = \alpha = r = 100$ under a noise-free condition and $h = 50$ms. 118
62 Overview of the MATLAB toolbox 140
63 Digitizing $\sin(t)$ 141

List of Tables

1 Fundamental operators associated with each implicit/semi-implicit scheme 51
2 Constant parameters of the exact differentiators 58
3 Parameters of the differentiators obtained from the tuning procedure 59
4 Results for the first-order differentiation under a noise-free condition 62
5 Results for the first-order differentiation in the presence of white noise 67
6 Parameters of the differentiators obtained from the tuning procedure 73
7 Results under a sinusoidal noise. Conditions are mentioned in Table 6. 76
8 Parameters of the differentiators obtained from the tuning procedure 87
9 Results under a bell-shaped noise. 90
10 Parameters of the differentiators obtained from the tuning procedure 92
11 Results under quantization 95
12 Parameters of the differentiators obtained from the tuning procedure 97
13 Parameters of the differentiators obtained from the tuning procedure 104
14 Parameters of the differentiators obtained from the tuning procedure 112
15 Results under oversized gains. 119
16 Effect of the solver's accuracy on the I-AO-STD with Newton's solver. 122
17 Effect of the solver's accuracy on the I-AO-STD with Householder's (Halley's) solver. 122

18 Effect of the solver's accuracy on the I-AO-STD with Bairstow's solver 123
19 A guidance to select the most appropriate differentiation method. The characters 2, ∞, V, T and C stand for $\bar{L}_2$, L_∞, VAR, THD and the calculation time, respectively. Colors blue and red show the best and the worst performances, respectively. 126
20 Parameters of the differentiators obtained from the tuning procedure 132
21 Results for the first-order differentiation with white noise 133
22 Parameters of the differentiators obtained from the tuning procedure 135
23 Results for the first-order differentiation with white noise 136

Nomenclature

AO-STD	arbitrary-order super-twisting differentiator
E-AO-STD	explicit arbitrary-order super-twisting differentiator
E-GHDD	explicit generalized homogeneous discrete-time differentiator
E-HDD	explicit homogeneous discrete-time differentiator
E-QD	explicit quadratic differentiator
E-STD	explicit super-twisting differentiator
E-STDAC	explicit super-twisting differentiator with adaptive coefficients
E-URED	explicit uniform robust exact differentiator
FFT	fast Fourier transform
GE	generalized equation
GHDD	generalized homogeneous discrete-time differentiator
HD	homogeneous differentiator
HDD	homogeneous discrete-time differentiator
HGD	high-gain differentiator
I-AO-FDFF	implicit arbitrary-order differentiator with first-order sliding-mode filtering
I-AO-STD	implicit arbitrary-order super-twisting differentiator
I-FDFF	first-order differentiator with first-order sliding-mode filtering
I-GHDD	implicit generalized homogeneous discrete-time differentiator
I-HDD	implicit homogeneous discrete-time differentiator
I-QD	implicit quadratic differentiator
I-STD	implicit super-twisting differentiator
I-URED	implicit uniform robust exact differentiator
LF	linear filter
ODE	ordinary differential equation
QD	quadratic differentiator
SI-AO-STD	semi-implicit arbitrary-order super-twisting differentiator
SI-URED	semi-implicit uniform robust exact differentiator
SI-STD	semi-implicit super-twisting differentiator
SMC	sliding-mode control
SHMD	Slotine-Hedrick-Misawa differentiator
SMB	Sliding-mode based
SNR	signal-to-noise ratio
STD	super-twisting differentiator
STDAC	super-twisting differentiator with adaptive coefficients
THD	total harmonic distortion
URED	uniform robust exact differentiator
VGED	variable gain exponent differentiator

Notation and definitions:

Definition 1 *The set-valued signum function is defined as follows:*

$$\operatorname{sgn}(s) = \begin{cases} -1 & \text{for} \quad s \in \mathbb{R}^- \\ [-1,+1] & \text{for} \quad s = 0 \\ +1 & \text{for} \quad s \in \mathbb{R}^+. \end{cases} \tag{1}$$

The inverse of the set-valued signum function ($y \in \operatorname{sgn}(x) \Leftrightarrow x \in \mathcal{N}_{[-1,1]}(y)$ for all x and y) can be obtained using the normal cone:

$$\mathcal{N}_{[-1,+1]}(s) = \begin{cases} \mathbb{R}^- & \text{for} \quad s = -1 \\ 0 & \text{for} \quad s \in [-1,+1] \\ \mathbb{R}^+ & \text{for} \quad s = +1. \end{cases} \tag{2}$$

Throughout the manuscript, I_d denotes the identity matrix, $\lceil x \rfloor^n = |x|^n \operatorname{sgn}(x)$, $\operatorname{diag}(a_0, \ldots, a_n)$ represents a diagonal matrix with elements $(a_0, \ldots, a_n)$, $\mathbb{R}_+^*$ ($\mathbb{R}_+$) denotes positive (non-negative) real numbers, and the set of complex numbers is denoted by $\mathbb{C}$. The signal $f(\cdot)$ is Lipschitz with constant L if $|f(t_2) - f(t_1)| \leq L|t_2 - t_1|$ for all $t_1, t_2 \in \mathbb{R}^+$, which holds if the first derivative of $f(\cdot)$ is bounded by L, i.e., $|\dot{f}(t)| < L$. Considering $G : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ a set-valued mapping, and $f : \mathbb{R}^n \to \mathbb{R}^m$ a single valued mapping, then $0 \in G(\cdot) + f(\cdot)$ is a generalized equation. In all rigor, we should denote (in all the generalized equations) the singletons as $\{\cdot\}$, however for convenience and in order to lighten the notation they will be simply denoted without the brackets. C^k is the space of at least k times differentiable functions, with continuous kth derivative. Thus, C^0 are continuous functions, C^1 are differentiable functions with continuous first-order derivative, and so on. Derivation order i of a continuous-time signal $f(t)$ is denoted by $f^{(i)}(t)$. Furthermore, the discrete-time counterparts of $f(t)$ and $f^{(i)}(t)$ at $t = t_k$ are denoted as $f_k = f(t_k)$ and $f_k^{(i)} = f^{(i)}(t_k)$, respectively. The notation $\sigma_{i,k}$ denotes the i-th element of the vector σ_k. Furthermore, I_d stands for the identity function, i.e., $x \mapsto x$, and $(\cdot)^{-1}$ stands for the inverse of mapping, possibly set-valued.

Throughout the manuscript, colors are used to provide extra information. In tables, red, black and blue indicate the worst, moderate and the best performances, respectively. Furthermore, in equations, red and blue indicate explicit and implicit parts, respectively.

It should be understood that all the considered signals $f_0(\mathrm{t})$ to be differentiated, are assumed to be differentiable up to the necessary orders. In other words, we do not aim at differentiating non-differentiable signals (like discontinuous signals, for instance). Furthermore, this study only considers the differentiators with integer orders. Thus, and fractional-order differentiators are out of the scopes of this research.

1 Introduction

Real-time differentiation is a well-known problem in Automatic Control and Signal Processing due to its great number of applications. Fig. 1 shows a typical application of a differentiator in a control loop. As can be seen, the controller usually requires the differentiations of the output ($f_0(t)$). Therefore, a differentiator needs to be utilized to receive the measurement through the feedback path $f(t) = f_0(t) + n(t)$ which is contaminated by an additive noise $n(t)$. Note that the controller needs the differentiations of $f_0(t)$ and not of $f(t)$. So, the differentiator needs to distinguish between the signal and the noise. This makes the differentiation a challenging problem because the noise may present stochastic characteristics.

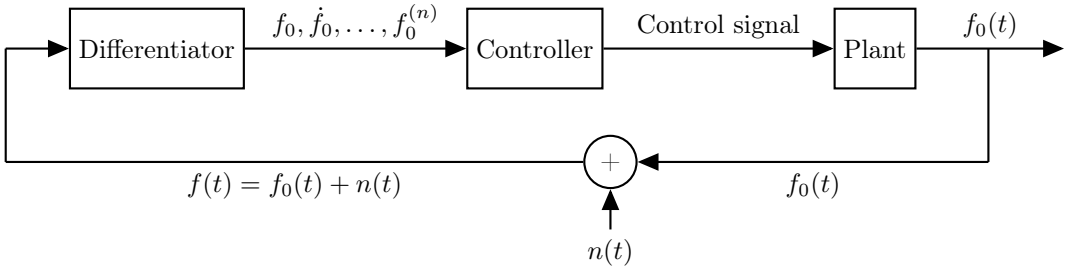

Figure 1: A typical application of a differentiator in a control loop.

Differentiators are traditionally designed in the frequency domain. In a noise-free case, the transfer function of the ideal differentiator is $T(s) = s$, where $s \in \mathbb{C}$ is the Laplace variable. Assuming that $x(t)$ is a noise-free input signal, the ideal differentiator would be $[y(t)] = s[x(t)]$. Time-discretization of this differentiator leads to the following equation which is called the Euler differentiator,

$$y_k = \frac{z-1}{hz} x_k \qquad \Longleftrightarrow \qquad y_k = \frac{x_k - x_{k-1}}{h}, \tag{3}$$

where $y_k = y(t_k)$, t_k are the discrete instants, with $t_{k+1} - t_k = h$, $h > 0$ is the sampling period, and z^{-1} is the unit delay operator. The main drawback of this differentiator is that it cannot distinguish between the signal and the noise. Thus, it calculates the differentiation of the noise as well. Since the noise usually exhibits high-frequency components, the Euler differentiator amplifies the effect of the noise. Thus, measurement noises with small magnitude can affect the signal differentiation significantly. This problem is usually resolved using a linear filter. In other words, a combination of a low-pass filter and the ideal differentiator is commonly used in practical applications. For example, a widely used differentiator is $T(s) = \dfrac{cs}{s+c}$, which is composed of a first-order low-pass filter $\dfrac{c}{s+c}$ and the ideal differentiator s. Time-discretization of this differentiator leads to the following equation:

$$y_k = \frac{y_{k-1} + c(x_k - x_{k-1})}{1 + hc}. \tag{4}$$

It can be seen that for $c \to \infty$, this differentiator tends to the Euler differentiator in (3). With small values of c, the effect of the differentiation at the previous time-step (y_{k-1}) on the current differentiation (y_k) will be more than the effect of the signal changes ($x_k - x_{k-1}$). As a result, the differentiator will be more robust to

high-frequency input noise and also present more phase-lag. Therefore, it can be concluded that a trade-off has to be made between the noise-filtration and phase-lag of a linear filter. In summary, the main drawbacks of these filters are as follows:

- Tuning the parameters of traditional filters could be challenging because these parameters should be tuned based on the noise specifications.
- Finite-time convergence of linear filters cannot be assured.
- A trade-off has to be made between the noise filtration performance and the phase-lag of the filtration stage. In other words, to improve the noise filtration performance of a linear filter (LF), it is necessary to decrease its cut-off frequency[1]. This increases the phase-lag of the filtration stage. This phase-lag always has an adverse effect on the control loops.

To alleviate these drawbacks, heuristic algorithms are proposed to tune the parameters of the linear filters [1, 2]. However, some of the mentioned limitations, e.g., the phase-lag, are intrinsic and cannot be solved by tuning the parameters. Moreover, in general, the optimality of these kinds of heuristic algorithms cannot be ensured. As a result, alternative methods have been introduced in the literature to solve the drawbacks of the aforementioned methods, e.g., Luenberger observer [3], high-gain differentiator (HGD) [3], ALIEN differentiator [4, 5], homogeneous differentiator (HD) [6–10], non-homogeneous differentiator [11], sliding-mode differentiators [12–17], kernel-based approach [18], etc.

The effect of the gain on HGDs has been studied in [19], and it is concluded that increasing the gain can increase the exactness of the HGD and its sensitivity to noise as well, and vice versa. So a trade-off has to be made between the exactness and the robustness to noise (this is in fact the case for all differentiators). Comparison of the HGD and E-AO-STD (see (7) and (16) below) through numerical simulations in [19] shows a slight advantage for the HGD. However, explicit discretization of AO-STD can affect the performances by imposing same numerical chattering and should be avoided. Alternative discretization schemes will be proposed in this work to solve this issue.

The comparison of some of these differentiators is addressed in [18, 20–24]. Comparisons between a sliding-mode differentiator, kernel-based method, HGD, ALIEN differentiator, extended Kalman filter and homogeneous differentiator have been presented using numerical simulations [18] and practical experiments [20, 22]. However, the main problem of these comparisons is that the influence of the discretization of the algorithms has not been addressed. Thus, it seems that sliding-mode-based differentiators are implemented in an explicit way which leads to unfair comparisons in [18, 22]. Moreover, as it is mentioned in [21], the ALIEN differentiator is not designed appropriately for the comparisons presented in [20].

Before reviewing the literature, it is necessary to present the following definitions:

[1]Considering ω as the input frequency of a low-pass filter, ω_c is defined as the cut-off frequency of the filter if for $\omega > \omega_c$ the gain of the filter is less than -3dB.

- **Exact differentiation**: Consider the input of a differentiator as $f(t) = f_0(t) + n(t)$, where $f_0(t)$ is the signal to be differentiated and $n(t)$ is the noise. A differentiator is called exact on some inputs if the output coincides with the derivative of $f_0(t)$.
- **Differentiator order**: A differentiator with the order n can calculate the n-th order differentiation of the input. While some differentiators have multiple outputs and can calculate the lower-order differentiations at the same time as well, such as the homogeneous differentiator, others cannot calculate the differentiations with the order lower than n at the same time, e.g., ALIEN differentiator.
- **Robustness to noise**: The input signal of differentiators is usually polluted by high-frequency noise. Differentiation usually amplifies the effect of this noise. Therefore, an appropriate differentiator should present the ability of noise filtration.
- **Robustness to disturbances**: Here, the disturbance is defined as sudden changes in input. Considering the existence of error in initial conditions of a differentiator, a more robust differentiator (with respect to the disturbance) presents a better transient response.

The following problems are usually considered as general drawbacks of differentiators [12, 13, 15]:

- Measurement noise and sampling period can affect the exactness of any differentiator.
- There is always a trade-off between the exactness and the robustness of a differentiator. In other words, exactness destroys the robustness.
- In many applications, it is necessary to utilize some noise filtering elements before a differentiator block. This pre-filtration imposes a phase-lag which affects the exactness of the differentiators.

The remainder of this manuscript is structured as follows. Several continuous-time differentiators are presented in Section 2. Afterward, the discretization of these differentiators is addressed in Section 3. The tuning procedure of the differentiators is presented in Section 4. Open-loop analysis of the differentiators is presented in Section 5, and finally, general conclusions are provided in Section 6. This manuscript is accompanied by a numerical simulation toolbox that will be introduced in Appendix E. In addition, several technical results are presented in Appendices A, B and D to F.

2 Continuous-time differentiators

Differentiators are mainly designed in the continuous-time domain. Several well-known continuous-time differentiators are introduced in this section.

2.1 Slotine-Hedrick-Misawa set-valued differentiator

Slotine-Hedrick-Misawa differentiator (SHMD) [25] was probably the first generation of sliding-mode based differentiators. This algorithm uses discontinuous functions to provide a sliding regime and therefore force the tracking error to zero. As a result, it is expected that the output converges to the real differentiation[1]. However, as it will be seen in Section 3, special attention should be taken into account for the discretization of the discontinuous functions to avoid the chattering. The general form of the SHMD is given as follows [25]:

$$\begin{cases} \dot{z}_i(t) \in z_{i+1}(t) - \alpha_i \Psi(\sigma_0(t)) - \kappa_i \sigma_0(t), & i = 0, \dots, n-1 \\ \dot{z}_n(t) \in -\alpha_n \Psi(\sigma_0(t)) - \kappa_n \sigma_0(t), \end{cases} \tag{5}$$

where $\sigma_0(t) = z_0(t) - f(t)$ is the sliding variable, $f(t)$ is the input of the differentiator, α_i and κ_i, $i = 0, 1, \dots, n$ are positive constants, n is the order of the differentiator, $\Psi(\cdot)$ is a set-valued function and $\in$ is written instead of $=$ to indicate the inclusion. The set-valued function $\Psi(\cdot) = \text{sgn}(\cdot)$ may be selected for this differentiator.

To analyse the properties of the differentiators, e.g., convergence, stability, ..., the problem of differentiator design is usually considered as the observer design for the system in the chain of integrators form. The existence of the sliding-phase and the behavior of the system in the reaching phase for the SHMD is studied in [25].

2.2 Super-twisting differentiator

Super-twisting differentiator (STD)[1] was developed based on the variable structure theory [12]. This differentiator is designed according to the ST algorithm as follows:

$$\begin{cases} \dot{z}_0(t) = -\lambda_0 L^{\frac{1}{2}} \lceil \sigma_0(t) \rfloor^{\frac{1}{2}} + z_1(t) \\ \dot{z}_1(t) \in -\lambda_1 L \,\text{sgn}(\sigma_0(t)), \end{cases} \tag{6}$$

where $\sigma_0(t) = z_0(t) - f(t)$, $f(t) = f_0(t) + n(t)$ is the input of the differentiator, $f_0(t)$ is the base signal which is polluted by noise $n(t)$, $z_0(t), z_1(t)$ are estimations for the input signal $f_0(t)$ and its first derivative $\dot{f}_0(t)$, respectively, λ_0 and λ_1 denote differentiation gains which may be designed according to [12], and L is the Lipschitz constant of $\dot{f}_0(t)$, i.e., $|\ddot{f}_0(t)| < L$.

Recent studies mainly dealt with the analysis of the convergence and optimality of exact differentiators. For example, optimality of the STD has been studied in [26], optimality of the STD in the presence of the large Gaussian white noise has been addressed in [27], and a new sliding-mode differentiator with exponential convergence rate has been proposed in [28]. Also, a family of smooth explicit Lyapunov functions has been

[1] Because of this property, sliding-mode based differentiators are termed "exact differentiators", even if they do not necessarily converge to the real differentiation.

[1] In some resources, STD is called Levant's differentiator because this differentiator has been introduced by Arie Levant in [12] for the first time.

proposed for the STD which allows to study the convergence and robustness of the differentiator with respect to initial condition [29].

2.3 Arbitrary-order super-twisting differentiator

The STD in (6) is basically a first-order differentiator. Therefore, a number of this type of first-order differentiators could be utilized in a cascade configuration to compute further derivatives. However, the cascade configuration of first-order differentiators is cumbersome and is not effective [30] because each differentiation stage amplifies the output chattering of the previous stage (see section 5.6 below for an analysis of cascade differentiation). In order to solve this drawback, an arbitrary-order super-twisting differentiator (AO-STD) has been proposed in [14]. Non-recursive AO-STD can be formulated as follows [31]:

$$\begin{cases} \dot{z}_i(t) = -\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_0(t) \rfloor^{\frac{n-i}{n+1}} + z_{i+1}(t), \quad i = 0, \dots, n-1 \\ \dot{z}_n(t) \in -\lambda_n L \operatorname{sgn}(\sigma_0(t)), \end{cases} \tag{7}$$

where $\sigma_0(t) = z_0(t) - f(t)$, and $f^{(n+1)}(t) \in [-L, L]$ for some $L > 0$. The parameters λ_i can be calculated through numerical simulations, see [9, 10, 14, 31]. Similarly, $f(t) = f_0(t) + n(t)$ is the input signal of the differentiator, and z_i is the estimation of $f_0^{(i)}(t)$. Additional filtration stages can also be taken into account in (7) [31]. It can be seen that (7) is a generalized form of the STD.

The global finite-time stability of this differentiator has been addressed in [29, 32] through explicit Lyapunov functions. Moreover, the following theorem is presented to show the robustness of the AO-STD (and also STD) against the noise in the continuous-time setting.

Theorem 1 ***[12, 33]*** *Let the input noise satisfy the inequality $|n(t)| \leq \epsilon$ for some $\epsilon > 0$. Then $|z_i(t) - f_0^{(i)}(t)| \leq \mu_i \epsilon^{(n-i+1)/(n+1)}$, $i = 0, \dots, n$, is established in finite-time for the AO-STD (and STD) in (6) and (7), where $\mu_i > 0$.*

2.4 Uniform robust exact differentiator

In the previous differentiators, the convergence time tends to infinity when the norm of the initial conditions of the differentiation error grows unboundedly. In order to solve this drawback, a modified STD called uniform robust exact differentiator (URED) has been developed in [34] as follows (higher-degree terms are shown in red):

$$\begin{cases} \dot{z}_0(t) = -\lambda_0 L^{\frac{1}{2}} \left(\lceil \sigma_0(t) \rfloor^{\frac{1}{2}} + \mu \lceil \sigma_0(t) \rfloor^{\frac{3}{2}} \right) + z_1(t) \\ \dot{z}_1(t) \in -\lambda_1 L \left(\frac{1}{2} \operatorname{sgn}(\sigma_0(t)) + 2\mu\sigma_0(t) + \frac{3}{2} \lceil \mu\sigma_0(t) \rfloor^2 \right), \end{cases} \tag{8}$$

The notation is similar to the above one, and $\mu \in \mathbb{R}_+^*$. In this method, high-degree terms are used in the mathematical equations to ensure the uniform convergence of the differentiator, i.e., the convergence speed does not depend on initial conditions. Some propositions and theorems have been presented in [34] to show

the finite-time convergence of this differentiator based on the Lyapunov theory. Note that the URED was originally introduced for $L = 1$ and the tuning parameter L is considered in this work to provide a unified presentation among all SMB differentiators. Application of this differentiation method for medical purposes was further addressed in [35]. Note that for $\mu = 0$, the URED becomes the STD.

2.5 Quadratic sliding-mode differentiator

In order to filter the input noise more efficiently, another type of exact differentiator called the quadratic differentiator (QD) has been introduced [36]. This type of differentiator has been utilized for removing white and impulsive types of noise [36]. This differentiator was further modified in [37] to improve its transient response by decreasing its overshoots. This modified QD is as follows:

$$\begin{cases} \dot{z}_0(t) = z_1(t) \\ \dot{z}_1(t) \in \begin{cases} -\alpha F \operatorname{sgn}(\sigma(t)) & \text{if} \quad \sigma(t) z_1(t) > 0 \\ -F \operatorname{sgn}(\sigma(t)) & \text{if} \quad \sigma(t) z_1(t) < 0 \end{cases} \\ \sigma(t) = 2F(z_0(t) - f(t)) + |z_1(t)| z_1(t), \end{cases} \tag{9}$$

where $F > 0$ and $\alpha > 0$ are gains to be tuned. As before, $f(t) = f_0(t) + n(t)$ is the input which is technically a base signal $f_0(t)$ polluted by an additive noise $n(t)$, $z_0(t)$ is an estimation of $f_0(t)$, $\sigma(t)$ is the sliding variable, and $z_1(t)$ is the estimation of the real differentiation, i.e., $z_1(t) \to \dot{f}(t)$ (output of the differentiator). The sliding variable of this differentiator has been modified in [38] to improve its convergence rate.

2.6 Homogeneous differentiator

A more general class of SMB differentiators called homogeneous differentiators[1] (HD) has been introduced [39]. Many differentiators such as high-gain and SMB differentiators can be formulated in the homogeneous form. This allows studying the convergence, robustness and performance characteristics in the same framework [20]. Homogeneous systems have important properties. For example, local stability implies global stability. Moreover, negative homogeneous degree implies finite-time convergence [6–8, 32, 40–42]. Based on the HDs, a differentiation toolbox has been introduced [7] which deals with the implementation of the AO-STD with high-degree linear terms [33]. Subsequently, an improved discretization method of the HDs was introduced [41] which led to a newer version of the differentiation toolbox [8]. Throughout this manuscript, the term homogeneous differentiator, or HD[2], refers to the method which was presented in [8, 41].

2.7 Adaptive differentiators

Some adaptation laws have been developed for the exact differentiators to tune their parameters automatically. In this context, two different adaptation techniques namely *adaptive coefficients* and *adaptive*

[1] Note that STD and AO-STD also belong to the class of homogeneous differentiators.

[2] The discretization method which is used in the HD is also called the matching discretization approach.

exponents have been introduced. In [43–45], adaptive laws are provided for the coefficients, while in other studies [46–48], the adaptation mechanisms are considered for the exponents. These schemes are introduced in Sections 2.7.1 and 2.7.2, respectively.

2.7.1 Adaptation mechanism for the coefficients

One of the latest adaptation mechanisms for the coefficients has been developed in [43], where the following adaptive differentiator has been proposed. Note that this differentiator is termed super-twisting differentiator with adaptive coefficients (STDAC). It reads as:

$$\begin{cases} \dot{z}_0(t) = -\lambda_0 \gamma(t) \lceil \sigma_0(t) \rfloor^{\frac{1}{2}} + z_1(t) & \text{(10a)} \\ \dot{z}_1(t) \in -\lambda_1 \gamma^2(t) \operatorname{sgn}(\sigma_0(t)), & \text{(10b)} \end{cases}$$

where $\sigma_0(t) = z_0(t) - f(t)$. It can be seen that for $\gamma(t) = \sqrt{L}$, (10) leads the standard STD (6). The adaptation law $\gamma(t)$ is obtained as follows:

$$\dot{\gamma}(t) = \frac{\gamma(t)}{2}\alpha \begin{cases} |\sigma_0(t)|^{-\frac{1}{2}} & \text{for} \quad |\sigma_0(t)| \geq 1 \\ |\sigma_0(t)| & \text{for} \quad |\sigma_0(t)| < 1 \\ \frac{1}{\gamma} - 1 & \text{for} \quad |\sigma_0(t)| < 1.1\epsilon, \end{cases} \tag{11}$$

where $\lambda(0) = 1$, $0 < \alpha < \lambda_0$, and ϵ is a design constant which is selected based on the amplitudes of the chattering and noise. According to (11), during the reaching-phase, $\gamma \to \infty$ to improve the exactness as well as the transient response. On the other hand, during the sliding-phase ($\sigma_0(t) = 0$), $\gamma \to 1$ to attenuate the chattering and improve the robustness to noise.

2.7.2 Adaptation mechanism for the exponents

The idea of variable gain exponent comes from the observation that by changing the exponent of an exact differentiator, a trade-off can be made between the exactness and robustness to noise. Comparisons, based on laboratory set-ups, between the LF and the STD shows that the exact differentiators are more sensitive to measurement noise [46–48]. Consequently, a modified exact differentiator has been developed for noisy environments, where the exponent of the sliding variable can take a value between 0.5 (corresponding to the STD) and 1 (corresponding to a LF). It leads to a trade-off between the exactness and the robustness.

The most recent study on the variable gain exponent differentiator (VGED) has been conducted in [48]. The continuous-time VGED reads as:

$$\begin{cases} \dot{z}_0(t) = -\lambda_0 \mu |\sigma_0(t)|^{\alpha(t)} \operatorname{sgn}(\sigma_0(t)) + z_1(t) & \text{(12a)} \\ \dot{z}_1(t) = -\lambda_1 \alpha(t) \mu^2 |\sigma_0(t)|^{2\alpha(t)-1} \operatorname{sgn}(\sigma_0(t)) & \text{(12b)} \\ \dot{\gamma}(t) = -\tau\gamma(t) + \tau|f_f(t)| & \text{(12c)} \\ \alpha(t) = \frac{1}{2}\left(1 + \frac{\gamma^q}{\gamma^q + \epsilon}\right), & \text{(12d)} \end{cases}$$

where, as before, $\sigma_0(t) = z_0(t) - f(t)$, and $f(t) = f_0(t) + n(t)$ is the input signal which is polluted by an additive noise $n(t)$. The signal $f_f(t)$ is the filtered input which can be calculated as follows:

$$f_f(t) = \mathcal{L}^{-1}\left\{ \frac{\left(\frac{s}{\omega_c}\right)^4}{\left(\left(\frac{s}{\omega_c}\right)^2 + 0.7654\left(\frac{s}{\omega_c}\right) + 1\right)\left(\left(\frac{s}{\omega_c}\right)^2 + 1.8478\left(\frac{s}{\omega_c}\right) + 1\right)} F(s) \right\}, \tag{13}$$

where $\mathcal{L}$ denotes the Laplace operator, i.e., $F(s) = \mathcal{L}\{f(t)\}$, and ω_c is the cutoff frequency. In fact, (13) is a fourth-order high-pass Butterworth filter, and according to [48], it is used "to capture the magnitude of the high-frequency input signal $f(t)$". It is also mentioned in [48] that "γ is a first-order low-pass filter of $f_f(t)$". As it can be seen, VGED has six parameters that should be tuned: $\lambda_0, \lambda_1, \mu, \tau, q, \omega_c$. The following points are provided in [48] to tune these parameters [1]:

- λ_0 and λ_1 are designed such that the polynomial $x^2 + \lambda_0 x + \lambda_1$ has appropriate eigenvalues.
- μ should be designed large enough to improve the transient response of the system. However, increasing this parameter leads to a higher noise sensitivity, i.e., lower robustness.
- $\epsilon > 0$ is a constant parameter which is chosen such that $\alpha \in [0.5, 1]$.
- τ is designed such that the dynamics of γ are ten-times slower than the dynamics of z_0 and z_1. In other words, τ is designed based on the bandwidth of a specific application which may not be known.
- ω_c is the cutoff frequency of the high-pass filter. This frequency should be greater (at least five-times) than the frequency of the base signal $f_0(t)$.
- $q > 1$ denotes the power tuning parameter. This parameter should be designed based on $|f_f(t)|$. According to [48], "increasing ω_c allows to assign α to be closer to 0.5 (respectively to be closer to 1) when $|f_f(t)|$ is very small (respectively when $|f_f(t)|$ is very large)".

Among all the differentiators which are studied in this report, VGED possesses the largest number of parameters. Tuning these parameters might be challenging especially in closed-loop applications where the characteristics of the system, e.g., the bandwidth, are not known or depend on the controller. The adaptation mechanism of the VGED is explained as follows:

1. It is assumed that the frequency components of the signal and the noise are separated enough such that the high-pass filter only passes the components of the noise. Hence, $|f_f(t_k)|$ indicates the amplitude of noise at time-step t_k.
2. According to (12c), increasing the noise magnitude ($|f_f(t)|$) leads to increasing γ.

[1] The next six comments are quoted from [48].

3. From (12d), increasing γ causes α to increase. For $\gamma \to \infty$ one has $\alpha \to 1$. It indicates that by increasing the noise amplitude, the differentiator behaves as a linear filter. On the other hand, for a noise-free case, $|f_f(t)| = 0$. In this case, $\alpha \to \frac{1}{2}$ holds which indicates that the VGED behaves as the STD.

As it can be seen, the adaptation mechanism of the VGED is based on the fact that the frequency of noise is higher than that of the signal. However, for white noise, where the noise components spread over all frequency components, it is not clear how the adaptation mechanism will behave. To decrease the number of parameters, it will be assumed that $\epsilon = \frac{1}{\mu}$ (as is proposed in [48]), and λ_0 and λ_1 are presented in Table 2, respectively. In this case, the VGED only has four parameters to be tuned, i.e., μ, τ, ω_c, q.

2.8 ALIEN differentiator

The literature shows that there are still other types of differentiators whose structure and mechanism are different from the previously mentioned ones. ALIEN differentiator [4] is one of these methods which calculates an arbitrary-order differentiation based on the concept of annihilators. In fact, it calculates the differentiation using integration to attenuate the noise effect. This differentiator has been analyzed in [49]. According to [49], the general form of the ALIEN differentiator is as follows:

$$z^{(n)}(t) = \frac{(-1)^n \gamma_{\kappa,\mu,n}}{T^n} \int_0^1 \frac{d^n}{d\tau^n}\{\tau^{\kappa+n}(1-\tau)^{\mu+n}\} f(\tau T) d\tau, \tag{14}$$

where $f(t)$ is the input of the differentiator, $z^{(n)}(t)$ is the n-th order differentiation of $f(t)$, $\gamma_{\kappa,\mu,n} = \frac{(\kappa+\mu+2n+1)!}{(\kappa+n)!(\mu+n)!}$, n is the differentiation order, T is called the estimation window, κ and μ are two parameters which can be tuned based on the comments made in [49]. However, we will later follow a common procedure to tune the parameters of all the tested differentiators, with a unified approach.

While an n-th order AO-STD has multiple outputs and can calculate any differentiation with the order 0 to n, ALIEN differentiator is a single-input single-output system and can only calculate the n-th order differentiations. For example, consider an application where the differentiation orders 0 to 3 are required. As it can be seen from (7), the outputs of a third-order AO-STD would be $z_i, i = 0, \ldots, 3$, which corresponds to the zero, first, second and third-order differentiation. However, the ALIEN differentiator is only designed for a specific differentiation, and four different ALIEN differentiators have to be designed to calculate differentiations order 0 to 3. Note that zero-order differentiation is expected to be the recovered input signal ($f_0(t)$) which is polluted by additive noise ($f(t) = f_0(t) + n(t)$).

From (14), a first-order ALIEN differentiator is designed as follows:

$$\begin{cases} \dot{z}(t) = -\frac{\gamma_{\kappa,\mu,(n=1)}}{T} \int_0^1 \Big((\kappa+1)\tau^{\kappa}(1-\tau)^{\mu+1} - \tau^{\kappa+1}(\mu+1)(1-\tau)^{\mu}\Big) f(\tau T) d\tau \\ \gamma_{\kappa,\mu,(n=1)} = \frac{(\kappa+\mu+2+1)!}{(\kappa+1)!(\mu+1)!}. \end{cases} \tag{15}$$

First, second and third-order ALIEN differentiators are implemented in the numerical simulation toolbox which is developed in this project.

2.9 High-gain differentiator

The HGD is a special case of high-gain observers, which has been initially introduced in [3]. The design of this differentiator was further addressed in [19]. In this study, a third-order HGD will be considered as follows (see [20, 50]).

$$\begin{cases} \dot{z}_0(t) = -L\lambda_0 \lceil e(t) \rfloor + z_1(t) \\ \dot{z}_1(t) = -L^2\lambda_1 \lceil e(t) \rfloor + z_2(t) \\ \dot{z}_2(t) = -L^3\lambda_2 \lceil e(t) \rfloor + z_3(t) \\ \dot{z}_3(t) = -L^4\lambda_3 \lceil e(t) \rfloor \end{cases} \tag{16}$$

where the notation is as before, $e(t) = f(t) - z_0(t)$ and $\lambda_i, i = 0, \ldots, 3$ are given in Table 2. The only tunable parameter of the HGD (L), will be tuned to make a trade-off between the exactness and the robustness to noise. It can bee seen that for $\alpha = 1$, the VGED (12) also leads to the HGD. Obtaining the error band of the HGD in the presence of noise was the topic of [19], and it has been mentioned that for a nth order HGD, the following optimal gain minimizes the error band corresponding to the ith output.

$$\frac{1}{L^*} = \sqrt[n]{\frac{k}{n-i}} \sqrt[n]{\frac{Q_i ||\mu||_\infty}{P_i ||f^(n)(t)||_\infty}} \tag{17}$$

where L^* is the optimal gain, P_i and Q_i indicate the ith elements of vectors P and Q, respectively, and P and Q are calculated as follows;

$$P = \int_0^\infty |e^{A_c t} B| dt \qquad Q = \int_0^\infty |e^{A_c t} B_c| dt \tag{18}$$

where

$$A_c = \begin{bmatrix} -\lambda_0 & 1 & \ldots & \ldots & 0 \\ -\lambda_1 & 0 & 1 & \ldots & 0 \\ \vdots & & \ddots & \vdots & \\ -\lambda_{n-1} & \ldots & \ldots & 0 & 1 \\ -\lambda_n & \ldots & \ldots & \ldots & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \quad B_c = \begin{bmatrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \tag{19}$$

According to [19], the optimal gain L^* provides the following error band for the ith output:

$$b_i(L^*) = (Q_k ||\mu||_\infty)^{1-\frac{i}{n}} (P_i ||f^(n)(t)||_\infty)^{i/n} (\frac{n}{i} - 1)^{\frac{i}{n}} (\frac{n}{n-i}) + \delta \tag{20}$$

where δ is an arbitrarily small quantity and $1 \leq i \leq n-1$.

2.10 Differentiators in closed-loop systems

The combination of several controllers and differentiators has been considered in the literature. For example, it is mentioned that the combination of the STD and ST controller leads to a discontinuous implementation

and therefore this combination should be avoided [51]. Subsequently, a higher-order differentiator and the ST controller are combined to achieve a continuous control law (see also [52] for an application of AO-STD in a closed-loop system). However, without addressing the discretization issue, these results do not seem to be valid in practical implementations. Indeed, it is shown that with a proper discretization method, discontinuous control laws can also be implemented with a tolerable amount of chattering [37, 53–61].

3 Discrete-time exact differentiators

The literature shows that time-discretization[1] of differentiators has not been studied extensively, and there are still some research gaps that should be addressed in future works. Generally speaking, discretization approaches can be categorized into two main methods, namely explicit (forward) and implicit (backward). These methods will be considered in the sequel.

Throughout this manuscript, explicit discretization refers to the fixed-step forward Euler discretization. For example, explicit discretization of the ODE: $\dot{x}(t) = f(x(t))$ leads to $x_{k+1} = hf_k + x_k$, where $h > 0$ is the sampling time and k denotes the equivalent discrete-time variable for the continuous-time variable t. On the other hand, implicit discretization of the mentioned continuous-time ODE gives $x_{k+1} = hf_{k+1} + x_k$ (see [53–56, 58, 62]), and one has to use an iterative scheme (Newton-Raphson, Halley or Householder algorithms) to advance the scheme. As is well-known, implicit methods are more complex to implement than the explicit ones, but they yield algorithms with much better properties (stability, accuracy, finite-time convergence, etc.). This has been shown for the discrete-time SMC [37, 53–61, 63].

All the algorithms are initialized at $k = 0$ and hold for $h \geq 0$. The time-step is $h = t_{k+1} - t_k > 0$. Simulations are made on $[0, t_f]$, $t_k = kh$, with $t_f = 10$s.

3.1 Explicit discretization of the exact differentiators

In practice, differentiators are usually implemented with an explicit Euler method, due to its simplicity, and the fact that it boils down to a mere copy of the continuous-time algorithm (at least concerning its overall structure, but the output may differ a lot). Explicit discretization of (6) and (7) yields (21) and (22), respectively.

$$\begin{cases} z_{0,k+1} = -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} + hz_{1,k} + z_{0,k} \\ z_{1,k+1} \in -h\lambda_1 L \operatorname{sgn}(\sigma_{0,k}) + z_{1,k}, \end{cases} \tag{21}$$

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n-i}{n+1}} + hz_{i+1,k} + z_{i,k}, \quad i = 0, \ldots, (n-1) \\ z_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k}) + z_{n,k}. \end{cases} \tag{22}$$

[1] Time-discretization of sliding-mode controllers and observers is also called *emulation* in the literature. The purpose of this topic is to obtain a discrete-time form of the continuous-time system in order to preserve the continuous-time characteristics.

Note that explicit terms are shown in red. The explicit discretization of AO-STD (22) and STD (21) has been investigated in [14, 33, 64, 65], and the following inequality is obtained for the AO-STD to determine the maximum output error during the sliding-phase:

$$\begin{cases} |z_i - f_0^{(i)}| \leq \mu_i \rho^{n-i+1}, & i = 0, 1, \dots, n \\ \rho = \max\{h, \varepsilon^{1/(n+1)}\}, \end{cases} \tag{23}$$

where μ_i depends on the parameters of the differentiator, i is the differentiation-order and n is the order of the differentiator. It is assumed that the input is polluted by a Lebesgue-measurable additive noise $f(t) = f_0(t) + \tilde{n}(t)$, where $f_0(t)$ and $\tilde{n}(t)$ denote the base signal and noise, respectively, such that $|\tilde{n}(t)| < \varepsilon$. It is mentioned in [33] that the one-step Euler discretization destroys the homogeneity of the continuous-time arbitrary-order differentiator for $n > 1$. Thus, the inequality (23) may not be valid for the Euler explicit discretization. To solve this problem, higher-degree terms of the Taylor expansion are included in the discrete-time differentiator, and it is shown that the previously mentioned inequality is approximately valid for the new discretization method [33]. This discretization scheme is presented in (24) (higher-degree terms are shown in red) and is termed the homogeneous discrete-time differentiator (HDD):

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n-i}{n+1}} + \sum_{j=1}^{n-i} \frac{h^j}{j!} z_{j+1,k} + z_{i,k}, \quad i = 0, \dots, (n-1) \\ z_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k}) + z_{n,k}. \end{cases} \tag{24}$$

From (9) and (8), explicit discretizations of QD and URED are also presented in (25) and (26), respectively.

$$\begin{cases} z_{0,k+1} = h z_{1,k} + z_{0,k} \\ z_{1,k+1} = \begin{cases} -h\alpha F \operatorname{sgn}(\sigma_k) + z_{1,k} & \sigma_k z_{1,k} > 0 \\ -hF \operatorname{sgn}(\sigma_k) + z_{1,k} & \sigma_k z_{1,k} < 0 \end{cases} \\ \sigma_k = 2F(z_{0,k} - f_k) + |z_{1,k}| z_{1,k}, \end{cases} \tag{25}$$

$$\begin{cases} z_{0,k+1} = -h\lambda_0 \left(\lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} + \mu \lceil \sigma_{0,k} \rfloor^{\frac{3}{2}} \right) + h z_{1,k} + z_{0,k} \\ z_{1,k+1} \in -h\lambda_1 \left(\frac{1}{2} \operatorname{sgn}(\sigma_{0,k}) + 2\mu\sigma_{0,k} + \frac{3}{2}\mu^2 \lceil \sigma_{0,k} \rfloor^2 \right) + z_{1,k}. \end{cases} \tag{26}$$

The HDD (24) was further modified and termed the generalized homogeneous discrete-time differentiator (GHDD) in [6] as follows:

$$z_{k+1} = \Phi z_k + hP\Lambda(\sigma_{0,k}), \tag{27}$$

where

$$P = \left[\begin{array}{c|c} 1 & \mathbf{0}_{1\times n} \\ \hline \mathbf{0}_{n\times 1} & \tilde{P} \end{array}\right], \quad \tilde{P} = \begin{bmatrix} 1 & \alpha_{12}h & \dots & \alpha_{1n}h^n \\ 0 & 1 & \dots & \alpha_{2n}h^{n-1} \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} -\lambda_0 L^{\frac{1}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n}{n+1}} \\ -\lambda_1 L^{\frac{2}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n-1}{n+1}} \\ \vdots \\ -\lambda_{n-1} L^{\frac{n}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{n+1}} \\ -\lambda_n L \operatorname{sgn}(\sigma_{0,k}) \end{bmatrix}, \tag{28}$$

where $z_k = [z_{0,k}, z_{1,k}, \ldots, z_{n,k}]^T$, $\Phi = e^{Ah}$ and $\alpha_{ij} \in \mathbb{R}$ can be determined according to the following equation:

$$\Phi P = P A_d, \tag{29}$$

where $A_d = \boldsymbol{I} + Ah$. It is shown, in [6], that (29) always leads to unique solutions for α_{ij}

The matrix P adds nonlinear terms into (27). In [6], it is noted that by adding these nonlinear terms, the difference equation corresponding to the variable $\sigma_{0,k}$ will not contain any discontinuous terms. This may lead to digital chattering attenuation and improve the estimation accuracy of the observer. To investigate this point, the dynamics of the estimation error related to second-order HDD and GHDD with homogeneity degree $d = -1$ and without perturbation ($f^{(3)}(t) = 0$) are provided in (30) and (31), respectively.

$$\begin{cases} \sigma_{0,k+1} = -L^{\frac{1}{3}}\lambda_0 h \lceil \sigma_{0,k} \rfloor^{\frac{2}{3}} \operatorname{sgn}(\sigma_{0,k}) + h\sigma_{1,k} + \sigma_{0,k} + \dfrac{h^2}{2}\sigma_{2,k} & \text{(30a)} \\ \sigma_{1,k+1} = -L^{\frac{2}{3}}\lambda_1 h \lceil \sigma_{0,k} \rfloor^{\frac{1}{3}} \operatorname{sgn}(\sigma_{0,k}) + h\sigma_{2,k} + \sigma_{1,k} & \text{(30b)} \\ \sigma_{2,k+1} = -L\lambda_2 h \operatorname{sgn}(\sigma_{0,k}) + \sigma_{2,k}, & \text{(30c)} \end{cases}$$

$$\begin{cases} \sigma_{0,k+1} = -L^{\frac{1}{3}}\lambda_0 h \lceil \sigma_{0,k} \rfloor^{\frac{2}{3}} \operatorname{sgn}(\sigma_{0,k}) + h\sigma_{1,k} + \sigma_{0,k} + \dfrac{h^2}{2}\sigma_{2,k} & \text{(31a)} \\ \sigma_{1,k+1} = -L^{\frac{2}{3}}\lambda_1 h \lceil \sigma_{0,k} \rfloor^{\frac{1}{3}} \operatorname{sgn}(\sigma_{0,k}) + h\sigma_{2,k} - \dfrac{h^2}{2}|\sigma_{0,k}|^{-1}\sigma_{0,k} + \sigma_{1,k} & \text{(31b)} \\ \sigma_{2,k+1} = -L\lambda_2 h \operatorname{sgn}(\sigma_{0,k}) + \sigma_{2,k}. & \text{(31c)} \end{cases}$$

From (31), $\sigma_{0,k}$ can be obtained as follows:

$$\begin{aligned} \sigma_{0,k+2} = &-L^{\frac{1}{3}}\lambda_0 h \lceil \sigma_{0,k+1} \rfloor^{\frac{2}{3}} + h\sigma_{1,k} - L^{\frac{2}{3}}\lambda_1 h^2 \lceil \sigma_{0,k} \rfloor^{\frac{1}{3}} \\ &+h^2\sigma_{2,k} + L\lambda_2 \tfrac{h^3}{2} \operatorname{sgn}(\sigma_{0,k}) + \tfrac{h^3}{2}\sigma_{2,k} - L\lambda_2 \tfrac{h^3}{2} \operatorname{sgn}(\sigma_{0,k}) + \sigma_{0,k+1}. \end{aligned} \tag{32}$$

It can be seen that the discontinuous term $L\lambda_2 \frac{h^3}{2} \operatorname{sgn}(\sigma_{0,k})$ is cancelled in (32). However, from (31b) and (31c), the discontinuous term still affects $\sigma_{1,k}$ and $\sigma_{2,k}$, and both variables appear in (32), which may potentially lead to chattering. A third-order GHDD will be considered in the simulations as follows:

$$\begin{cases} z_{0,k+1} = z_{0,k} + hz_{1,k} + \dfrac{h^2}{2}z_{2,k} + \dfrac{h^3}{6}z_{3,k} - h\lambda_0 L^{\frac{1}{4}} \lceil \sigma_{0,k} \rfloor^{\frac{3}{4}} & \text{(33a)} \\ z_{1,k+1} = z_{1,k} + hz_{2,k} + \dfrac{h^2}{2}z_{3,k} - h\lambda_1 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} - \alpha_{12}h^2\lambda_2 L^{\frac{3}{4}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{4}} - \alpha_{13}h^3 L\lambda_3 \operatorname{sgn}(\sigma_{0,k}) & \text{(33b)} \\ z_{2,k+1} = z_{2,k} + hz_{3,k} - h\lambda_2 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} - \alpha_{23}h^2\lambda_3 L \operatorname{sgn}(\sigma_{0,k}) & \text{(33c)} \\ z_{3,k+1} = z_{3,k} - hL \operatorname{sgn}(\sigma_{0,k}). & \text{(33d)} \end{cases}$$

Substituting $n = 3$ into (28), yields

$$\begin{bmatrix} 1 & h & \alpha_{12}h^2 + \frac{h^2}{2} & \alpha_{13}h^3 + \frac{\alpha_{23}h^3}{2} + \frac{h^3}{6} \\ 0 & 1 & h + \alpha_{12}h & \alpha_{13}h^2 + \alpha_{23}h^2 + \frac{h^2}{2} \\ 0 & 0 & 1 & h + \alpha_{23}h \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & h & 0 & 0 \\ 0 & 1 & h + \alpha_{12}h & \alpha_{12}h^2 + \alpha_{13}h^2 \\ 0 & 0 & 1 & h + \alpha_{23}h \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{34}$$

Parameters $\alpha_{12} = -\frac{1}{2}, \alpha_{23} = -1$ and $\alpha_{13} = \frac{1}{3}$ satisfy (34). The explicit discretization of adaptive schemes can be derived based on the procedure which has been presented above. From (12), the explicit discretization of VGED yields:

$$\begin{cases} z_{0,k+1} = -h\lambda_0\mu|\sigma_{0,k}|^{\alpha_k}\,\mathrm{sgn}(\sigma_{0,k}) + hz_{1,k} + z_{0,k} & \text{(35a)} \\ z_{1,k+1} = -h\lambda_1\alpha_k\mu^2|\sigma_{0,k}|^{2\alpha_k-1}\,\mathrm{sgn}(\sigma_{0,k}) + z_{1,k} & \text{(35b)} \\ \gamma_{k+1} = -h\tau\gamma_k + h\tau|f_{f,k}| + \gamma_k & \text{(35c)} \\ \alpha_k = \frac{1}{2}\left(1 + \frac{\gamma_k^q}{\gamma_k^q + \epsilon}\right), & \text{(35d)} \end{cases}$$

3.2 Explicit HD

Another SMB differentiator termed homogeneous differentiator (HD) has been proposed [8, 41]. In this scheme, the differentiation problem is formulated as the following chain of integrator:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + e_{n+1}f^{(n+1)}(t) \\ y(t) &= e_1^T x(t) \end{aligned} \tag{36}$$

where, as before, $f(t)$ is the signal to be differentiated, $A = \begin{bmatrix} 0_{n\times 1} & I_{n\times n} \\ 0 & 0_{1\times n} \end{bmatrix}$, x is the vector of differentiations, and e_i indicates a column vector with ith element equal to one, and other elements equal to zero. Discrete-form of (36) has been given in [8, 41] as follows:

$$\begin{aligned} x_{k+1} &= \Phi x_k + hH_k \\ y_k &= e_1^T x_k \end{aligned} \tag{37}$$

where $\Phi = \exp(Ah)$, and $H_k = [\frac{h^n}{(n+1)!}f_k^{(n+1)}\ ,\ \frac{h^{n-1}}{n!}f_k^{(n+1)}\ ,\ \dots\ ,\ f_k^{(n+1)}]^T$. If the input signal is assumed to be Lipschitz continuous with constant L, one has $f_k^{(n+1)} \in [-L, L]$ [33]. In this case, it is true that $H_k \in [\frac{h^n}{(n+1)!}\ ,\ \frac{h^{n-1}}{n!}\ ,\ \dots\ ,\ 1]^T[-L, L]$. The following continuous-time state observer is proposed in [8, 41] for (36):

$$\dot{z} = Az + \Psi(\sigma_0)\sigma_0 \tag{38}$$

where $\Psi(\sigma_0) = [\Psi_0(\sigma_0)\ ,\dots,\Psi_n(\sigma_0)]^T$ and $\Psi_i(\sigma_0) = \lambda_i|\sigma_0|^{\frac{n-i}{n+1}}$. The dynamic of the differentiation error $\sigma = x - z$ can be obtained as follows:

$$\dot{\sigma} = [A - \Psi(\sigma_0)e_1^T]\sigma + e_{n+1}f^{(n+1)} \tag{39}$$

The eigenvalues of the matrix $[A - \Psi(\sigma_0)e_1^T]$ can be obtained as $s_i = p_i|\sigma_0|^{-\frac{1}{n+1}}$, where p_i are the roots of the polynomial $p^{n+1} + \lambda_0 p^n + \cdots + \lambda_{n-1}p + \lambda_n$.

The HD differentiator is proposed [8, 41] as follows:

$$z_{k+1} = \Phi z_k + \Lambda(\sigma_{0,k})\sigma_{0,k} \tag{40}$$

where $\Lambda(\sigma_{0,k}) = [\Lambda_0(\sigma_{0,k}), \dots, \Lambda_n(\sigma_{0,k})]^T$. The injection function $\Lambda(\sigma_{0,k})$ is designed such that the eigenvalues of the matrix $[\Phi - \Lambda(\sigma_{0,k})e_1^T]$ are located at $z_i = \exp(hs_{i,k}(\sigma_{o,k}))$ through the matching approach [8, 41], where $s_{i,k}(\sigma_{0,k}) = p_i|\sigma_{0,k}|^{-\frac{1}{n+1}}$.

Remark 1 *The strategy behind the HD is to obtain a pseudo-linear representation of the AO-STD and to use a pole placement strategy. Note that in HD, it is assumed that all eigenvalues are at the same place, i.e., $s_c = s_i, i = 1, \dots, n$. The robustness factor r which is used in the simulations refers to the eigenvalues of the continuous-time setting s_c.*

3.3 Implicit discretization of the exact differentiators

While several studies have been conducted on the implicit discretization of homogeneous systems and sliding-mode controllers [37, 53–56, 58, 59, 61, 62], implicit discretization of differentiators has not been addressed extensively. Implicit time-discretization of set-valued sliding-mode observers and differentiators can be traced back to [55], where embryonic analysis may be found. As far as the authors have investigated, only a few numbers of resources [16, 17, 37, 38, 59, 66, 67] have studied the implicit time-discretization for differentiators.

3.3.1 Implicit discretization of the super-twisting differentiator

From (6), the implicit discretization of the STD yields:

$$\begin{cases} z_{0,k+1} = -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k+1} \rfloor^{\frac{1}{2}} + hz_{1,k+1} + z_{0,k} & \text{(41a)} \\ z_{1,k+1} \in -h\lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) + z_{1,k}. & \text{(41b)} \end{cases}$$

where implicit terms are shown in blue. Substituting (41b) into (41a) yields the following generalized equation:

$$z_{0,k+1} \in -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k+1} \rfloor^{\frac{1}{2}} - h^2\lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) + hz_{1,k} + z_{0,k}. \tag{42}$$

Considering $z_{0,k+1} = \sigma_{0,k+1} + f_k$, (42) can be rewritten as

$$\sigma_{0,k+1} \in -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k+1} \rfloor^{\frac{1}{2}} - h^2\lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) + hz_{1,k} + z_{0,k} - f_k. \tag{43}$$

Equation (43) yields the following generalized equation[1]:

$$g(\sigma_{0,k+1}) \in -L\lambda_1 h^2 \operatorname{sgn}(\sigma_{0,k+1}), \tag{44}$$

[1]Note that this generalized equation has been recently solved in [58].

where

$$\begin{cases} g(x) = x + a\lceil x \rfloor^{1/2} + b_k \\ a = h\lambda_0 L^{\frac{1}{2}} > 0, \quad b_k = -z_{0,k} - hz_{1,k} + f_k \\ \xi(y) = g^{-1}(y) = \begin{cases} \left(\frac{-a+\sqrt{a^2-4(b_k-y)}}{2}\right)^2 & \text{if} \quad y \geq b_k \\ -\left(\frac{a-\sqrt{a^2+4(b_k-y)}}{2}\right)^2 & \text{if} \quad y < b_k, \end{cases} \end{cases} \tag{45}$$

which means that the scheme is advanced with the fundamental operator (46), which may be compared to the fundamental operator associated with implicit first-order SMC $x_{k+1} \mapsto \left(I_d + h\,\mathrm{sgn}(\cdot)\right)^{-1}(x_k)$ [61].

$$x \mapsto \left(I_d + a\lceil \cdot \rfloor^{\frac{1}{2}} + L\lambda_1 h^2\,\mathrm{sgn}(\cdot)\right)^{-1}(-b_k) \tag{46}$$

The generalized equation (44) and (45) can be solved based on the interpretation in Fig. 2 as follows:

- **Case 1:** $b_k < -h^2\lambda_1 L$

  Considering Fig. 2, $\sigma_{0,k+1} > 0$ is obtained for this case. Substituting $\mathrm{sgn}(\sigma_{0,k+1}) = 1$ in (42) yields

$$\sigma_{0,k+1} + a\sigma_{0,k+1}^{\frac{1}{2}} + h^2\lambda_1 L + b_k = 0. \tag{47}$$

  The solution of (47) is as follows:

$$\sqrt{|\sigma_{0,k+1}|} = \frac{-a + \sqrt{a^2 - 4(b_k + L\lambda_1 h^2)}}{2}. \tag{48}$$

- **Case 2:** $b_k \in [-h^2\lambda_1 L, h^2\lambda_1 L]$

  According to Fig. 2, $\sigma_{0,k+1} = 0$ is obtained for this case. Therefore, (43) gives

$$\begin{aligned} & b_k \in -h^2\lambda_1 L\,\mathrm{sgn}(0) = -h^2\lambda_1 L[-1,1] && \Leftrightarrow \\ & b_k = -h^2\lambda_1 L\xi \qquad \text{for some } \xi \in [-1,1] && \Rightarrow \\ & \xi = \frac{b_k}{-h^2\lambda_1 L}, \end{aligned} \tag{49}$$

  where ξ is called a selection of the set-valued signum function at zero. It is a fact that implicit methods allow to select automatically a suitable selection, while explicit ones cannot. This may be considered the source of numerical chattering in SMC [53]. Substituting (49) and $\sigma_{0,k+1} = 0$ into (41), the equations of the I-STD for this case are obtained as follows:

$$\begin{cases} z_{0,k+1} = hz_{1,k+1} + z_{0,k} & \text{(50a)} \\ z_{1,k+1} = z_{1,k} + \frac{b_k}{h} = -\frac{z_{0,k} - f_k}{h} = -\frac{\sigma_{0,k}}{h}. & \text{(50b)} \end{cases}$$

- **Case 3:** $b_k > h^2\lambda_1 L$

  In this case, $\sigma_{0,k+1} < 0$ is obtained from Fig. 2. Substituting $\text{sgn}(\sigma_{0,k+1}) = -1$ yields

$$\sigma_{0,k+1} - a|\sigma_{0,k+1}|^{\frac{1}{2}} - h^2\lambda_1 L + b_k = 0. \tag{51}$$

  The solution of (51) is as follows:

$$\sqrt{|\sigma_{0,k+1}|} = \frac{-a + \sqrt{a^2 + 4(b_k - L\lambda_1 h^2)}}{2}. \tag{52}$$

Substituting $\text{sgn}(\sigma_{0,k+1})$ and $\sqrt{\sigma_{0,k+1}}$, which are obtained for different cases, into (41), the flowchart of the I-STD is obtained as depicted in Fig. 3.

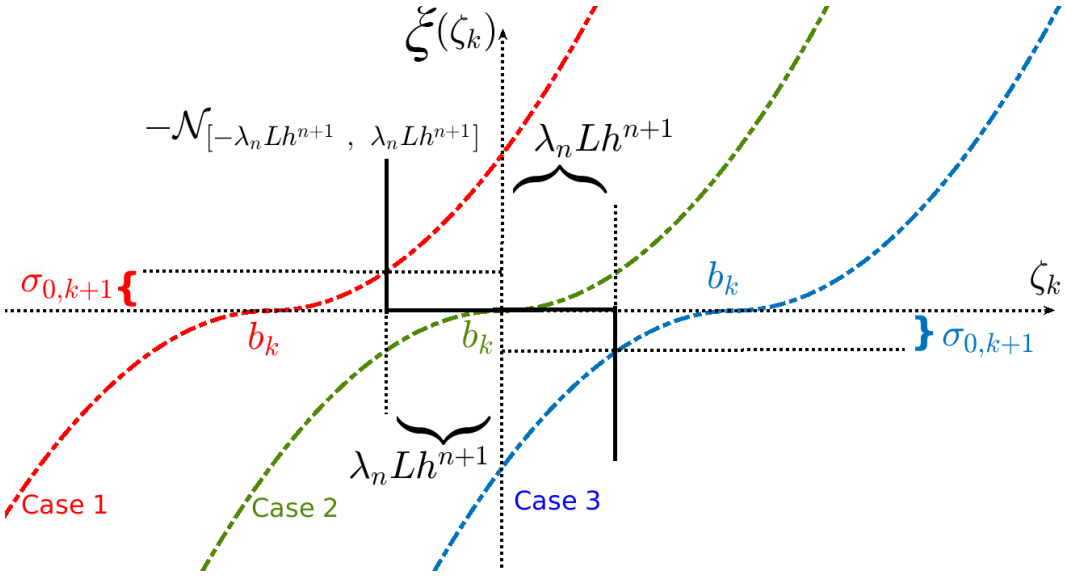

Figure 2: Graphical interpretation of (44): $\xi_k \in \text{sgn}(x_k) \Leftrightarrow x_k \in \mathcal{N}_{[-1,1]}(\xi_k)$.

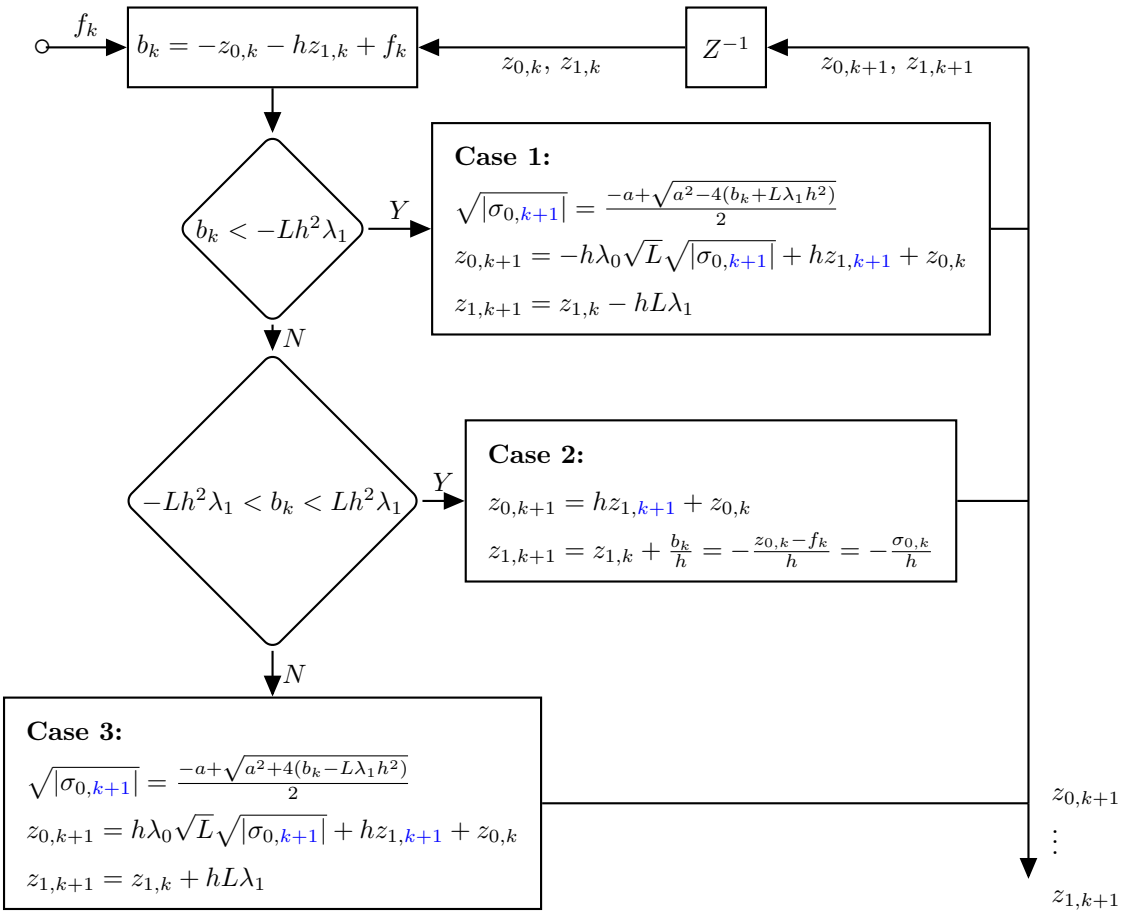

Figure 3: Flowchart of the I-STD. The block Z^{-1} indicates one-step delay.

3.3.2 Implicit discretization of the quadratic differentiator

Implicit discretization of the modified QD, in (9), has been presented in [37]. In this case, it is convenient to rewrite (9) as follows:

$$\begin{cases} \dot{z}_0 = z_1 \\ \dot{z}_1 = -F\text{gsgn}\big(\text{gsgn}(-\alpha, z_1, -1), \sigma, \text{gsgn}(1, z_1, \alpha)\big) \\ \sigma = 2F(z_0 - f) + |z_1|z_1, \end{cases} \tag{53}$$

where

$$\text{gsgn}(A, z, B) = \begin{cases} A & \text{if} \quad z < 0 \\ [\min(A \cup B), \max(A \cup B)] & \text{if} \quad z = 0 \\ B & \text{if} \quad z > 0. \end{cases} \tag{54}$$

Let us follow the steps in [37] to extract the modified QD in an implementable way. Applying the

backward Euler discretization on (53), yields for $k \geq 1$:

$$\begin{cases} \dfrac{z_{0,k} - z_{0,k-1}}{h} = z_{1,k} & \text{(55a)} \\ \dfrac{z_{1,k} - z_{1,k-1}}{h} = -F\,\mathrm{gsgn}\big(\mathrm{gsgn}(-\alpha, z_{1,k}, -1), \sigma_k, \mathrm{gsgn}(1, z_{1,k}, \alpha)\big) & \text{(55b)} \\ \sigma_k = 2F(z_{0,k} - f_k) + |z_{1,k}|z_{1,k}. & \text{(55c)} \end{cases}$$

Substituting (55a) into (55c) gives

$$\begin{cases} \dfrac{z_{1,k} - z_{1,k-1}}{h} = -F\,\mathrm{gsgn}\big(\mathrm{gsgn}(-\alpha, z_{1,k}, -1), \sigma_k, \mathrm{gsgn}(1, z_{1,k}, \alpha)\big) & \text{(56a)} \\ \sigma_k = |z_{1,k}|z_{1,k} + 2Fhz_{1,k} + 2F(z_{0,k-1} - f_k). & \text{(56b)} \end{cases}$$

Equation (56) can be simplified as follows:

$$\begin{cases} \dfrac{z_{1,k} - z_{1,k-1}}{h} = -F\,\mathrm{gsgn}\big(\mathrm{gsgn}(-\alpha, z_{1,k}, -1), z_{1,k} - z^*_{1,k}, \mathrm{gsgn}(1, z_{1,k}, \alpha)\big) & \text{(57a)} \\ z^*_{1,k} = \mathrm{sgn}(z_{0,k-1} - f_k)(Fh - \sqrt{F^2h^2 + 2F|z_{0,k-1} - f_k|}), & \text{(57b)} \end{cases}$$

where $z^*_{1,k}$ is the value of $z_{1,k}$ that satisfies $\sigma_k = 0$. Equation (56a) can be rewritten as:

$$-(z_{1,k} - z_{1,k-1}) = \mathrm{gsgn}(\mathrm{gsgn}(-\alpha hF, z_{1,k}, -hF), z_{1,k} - z^*_{1,k}, \mathrm{gsgn}(hF, z_{1,k}, \alpha hF)). \quad (58)$$

According to Proposition 1, presented in Appendix A, (58) can also be written as follows:

$$z_{1,k} = z_{1,k-1} - \mathrm{gsat}(\mathrm{gsat}(-\alpha hF, z_{1,k-1}, -hF), z_{1,k-1} - z^*_{1,k}, \mathrm{gsat}(hF, z_{1,k-1}, \alpha hF)). \quad (59)$$

Finally, the algorithm related to the implicit discretization of the modified QD in (53) is as follows:

$$\begin{cases} z^*_{1,k} = \mathrm{sgn}(z_{0,k-1} - f_k)(Fh - \sqrt{F^2h^2 + 2F|z_{0,k-1} - f_k|}) \\ z_{1,k} = z_{1,k-1} - \mathrm{gsat}(\mathrm{gsat}(-\alpha hF, z_{1,k-1}, -hF), z_{1,k-1} - z^*_{1,k}, \mathrm{gsat}(hF, z_{1,k-1}, \alpha hF)) \\ z_{0,k} = hz_{1,k} + z_{0,k-1}, \end{cases} \quad (60)$$

for all $k \geq 1$, and given initial data $z_{0,0} = z_{10}$, $z_{1,0} = z_{20}$, and u_1. Phase-portrait of the I-QD for $F = 100, \alpha = 5$ and input signal $\sin(t)$ is shown in Fig. 4 for $h = 1$ms. It can be seen that the estimation error of the input signal $(z_0 - f)$ and the estimation error of the first-order differentiation $(z_1 - df/dt)$ converge to the origin for 200 randomly selected initial conditions. Initial values for z_0 and z_1 are selected within the interval $[-10, 10]$.

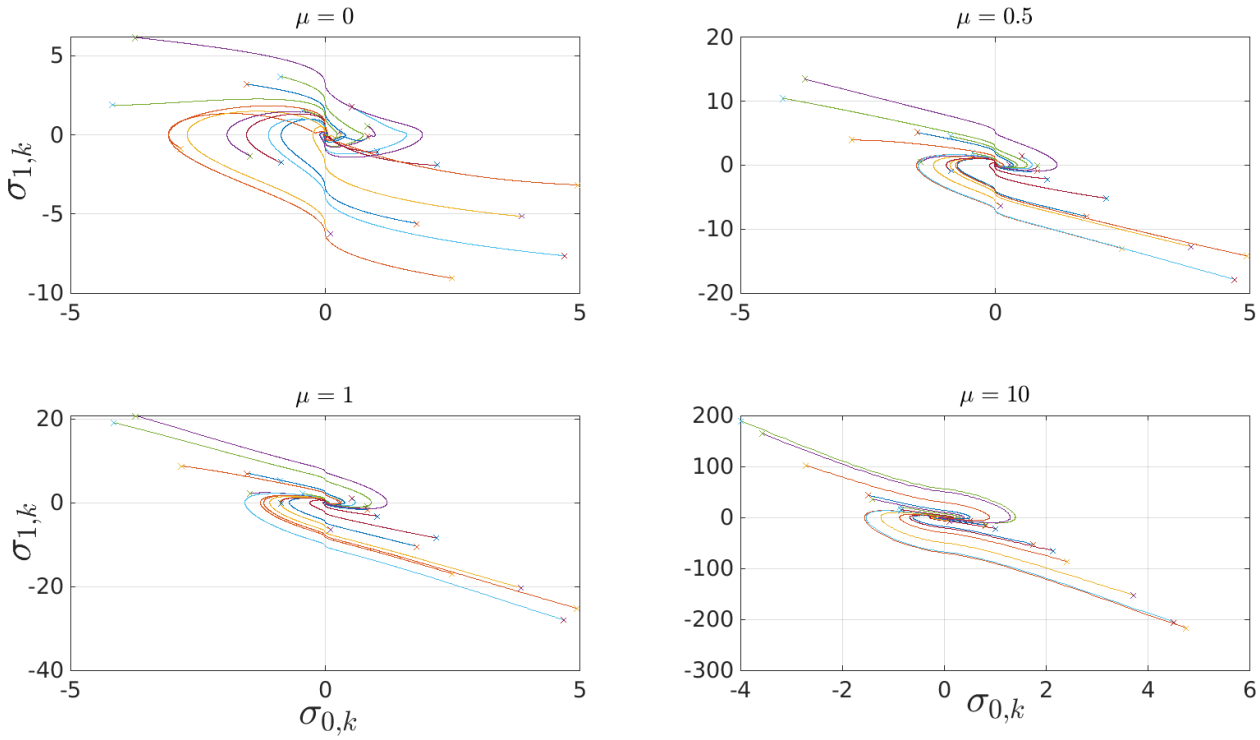

Figure 4: Phase-portrait of the I-QD. Input signal is $\sin(t)$. $F = 100, \alpha = 5, h = 1\text{ms}$.

3.3.3 Implicit discretization of the uniform robust exact differentiator

The implicit discretization of URED can be obtained in a similar manner as in Section 3.3.1. From (8), the implicit discretization of URED yields,

$$\begin{cases} z_{0,k+1} = -h\lambda_0 L^{\frac{1}{2}}\Big(\lceil \sigma_{0,k+1}\rfloor^{\frac{1}{2}} + \mu\lceil \sigma_{0,k+1}\rfloor^{\frac{3}{2}}\Big) + hz_{1,k+1} + z_{0,k} & \text{(61a)} \\ z_{1,k+1} \in -h\lambda_1 L\Big(\frac{1}{2}\operatorname{sgn}(\sigma_{0,k+1}) + 2\mu\sigma_{0,k+1} + \frac{3}{2}\mu^2\lceil \sigma_{0,k+1}\rfloor^2\Big) + z_{1,k}. & \text{(61b)} \end{cases}$$

Substituting (61b) into (61a) yields the following generalized equation:

$$z_{0,k+1} \in -h\lambda_0 L^{\frac{1}{2}}\Big(\lceil \sigma_{0,k+1}\rfloor^{\frac{1}{2}} + \mu\lceil \sigma_{0,k+1}\rfloor^{\frac{3}{2}}\Big) - h^2\lambda_1 L\Big(\frac{1}{2}\operatorname{sgn}(\sigma_{0,k+1}) + 2\mu\sigma_{0,k+1} + \frac{3}{2}\mu^2\lceil \sigma_{0,k+1}\rfloor^2\Big) + hz_{1,k} + z_{0,k}. \tag{62}$$

Substituting $z_{0,k+1} = \sigma_{0,k+1} + f_k$ and $b_k = -z_{0,k} - hz_{1,k} + f_k$ into (62) yields

$$\sigma_{0,k+1} \in -h\lambda_0 L^{\frac{1}{2}}\Big(\lceil \sigma_{0,k+1}\rfloor^{\frac{1}{2}} + \mu\lceil \sigma_{0,k+1}\rfloor^{\frac{3}{2}}\Big) - h^2\lambda_1 L\Big(\frac{1}{2}\operatorname{sgn}(\sigma_{0,k+1}) + 2\mu\sigma_{0,k+1} + \frac{3}{2}\mu^2\lceil \sigma_{0,k+1}\rfloor^2\Big) - b_k. \tag{63}$$

The fundamental operator stemming from (63) is given by

$$x \mapsto \Big(I_d + h\lambda_0 L^{\frac{1}{2}}\big(\lceil\cdot\rfloor^{\frac{1}{2}} + \mu\lceil\cdot\rfloor^{\frac{3}{2}}\big) + h^2\lambda_1 L\big(\frac{1}{2}\operatorname{sgn}(\cdot) + 2\mu(\cdot) + \frac{3}{2}\mu^2\lceil\cdot\rfloor^2\big)\Big)^{-1}(-b_k) \tag{64}$$

The generalized equation can be solved as follows:

- **Case 1:** $b_k < -\frac{h^2\lambda_1 L}{2}$

  Considering Fig. 2, $\sigma_{0,k+1} > 0$ is obtained for this case. Substituting $\text{sgn}(\sigma_{0,k+1}) = 1$ in (63) gives

$$\sigma_{0,k+1} + h\lambda_0 L^{\frac{1}{2}}\left(\sigma_{0,k+1}^{\frac{1}{2}} + \mu\sigma_{0,k+1}^{\frac{3}{2}}\right) + h^2\lambda_1 L\left(\frac{1}{2} + 2\mu\sigma_{0,k+1} + \frac{3}{2}\mu\sigma_{0,k+1}^2\right) + b_k = 0. \tag{65}$$

  Equation (65) can be rewritten as follows:

$$a_4\sigma_{0,k+1}^2 + a_3\sigma_{0,k+1}^{\frac{3}{2}} + a_2\sigma_{0,k+1} + a_1\sigma_{0,k+1}^{\frac{1}{2}} + a_0 = 0, \tag{66}$$

  with

$$\begin{cases} a_4x_k^4 + a_3x_k^3 + a_2x_k^2 + a_1x_k + a_0 = 0. \\ a_4 = \frac{3\lambda_1 Lh^2\mu}{2} \\ a_3 = h\lambda_0\mu L^{\frac{1}{2}} \\ a_2 = 1 + 2\mu\lambda_1 Lh^2 \\ a_1 = h\lambda_0 \\ a_0 = \frac{\lambda_1 Lh^2}{2} + b_k. \end{cases} \tag{67}$$

  Considering $\sigma_{0,k+1}^{\frac{1}{2}} \triangleq x_k$ gives

$$a_4x_k^4 + a_3x_k^3 + a_2x_k^2 + a_1x_k + a_0 = 0. \tag{68}$$

  This is a polynomial equation which can be solved by a suitable numerical method. Assuming that X_k is the solution of (68), one has $\sigma_{0,k+1} \triangleq X_k^2$.

- **Case 2:** $b_k \in [-\frac{h^2\lambda_1 L}{2}, \frac{h^2\lambda_1 L}{2}]$

  According to Fig. 2, $\sigma_{0,k+1} = 0$ is obtained for this case. Therefore, (63) gives

$$b_k = -\frac{h^2\lambda_1 L}{2}\,\text{sgn}(\sigma_{0,k+1}) \quad \Rightarrow \quad \text{sgn}(\sigma_{0,k+1}) = \frac{2b_k}{-h^2\lambda_1 L}. \tag{69}$$

  The term $\frac{2b_k}{-h^2\lambda_1 L}$ in (69) is called the selection of the set-valued signum function.

- **Case 3:** $b_k > \frac{h^2\lambda_1 L}{2}$

  Considering Fig. 2, $\sigma_{0,k+1} < 0$ is obtained for this case. Substituting $\text{sgn}(\sigma_{0,k+1}) = -1$ in (63) gives

$$\sigma_{0,k+1} - h\lambda_0 L^{\frac{1}{2}}\left(\sigma_{0,k+1}^{\frac{1}{2}} + \mu\sigma_{0,k+1}^{\frac{3}{2}}\right) - h^2\lambda_1 L\left(\frac{1}{2} - 2\mu\sigma_{0,k+1} + \frac{3}{2}\mu\sigma_{0,k+1}^2\right) + b_k = 0. \tag{70}$$

  Equation (70) can be rewritten as follows:

$$a_4\sigma_{0,k+1}^2 + a_3\sigma_{0,k+1}^{\frac{3}{2}} + a_2\sigma_{0,k+1} + a_1\sigma_{0,k+1}^{\frac{1}{2}} + a_0 = 0, \tag{71}$$

with

$$\begin{cases} a_4 x_k^4 + a_3 x_k^3 + a_2 x_k^2 + a_1 x_k + a_0 = 0. \\ a_4 = -\frac{3\lambda_1 L h^2 \mu}{2} \\ a_3 = -h\lambda_0 L^{\frac{1}{2}} \mu \\ a_2 = -(1 + 2\mu\lambda_1 L h^2) \\ a_1 = -h\lambda_0 L^{\frac{1}{2}} \\ a_0 = \frac{\lambda_1 L h^2}{2} + b_k. \end{cases} \tag{72}$$

Considering $x_k \triangleq (-\sigma_{0,k+1})^{\frac{1}{2}}$ gives

$$a_4 x_k^4 + a_3 x_k^3 + a_2 x_k^2 + a_1 x_k + a_0 = 0. \tag{73}$$

This is a polynomial equation which can be solved numerically. Assuming that X_k is the solution of (73), we have

$$\sigma_{0,k+1} = -X_k^2. \tag{74}$$

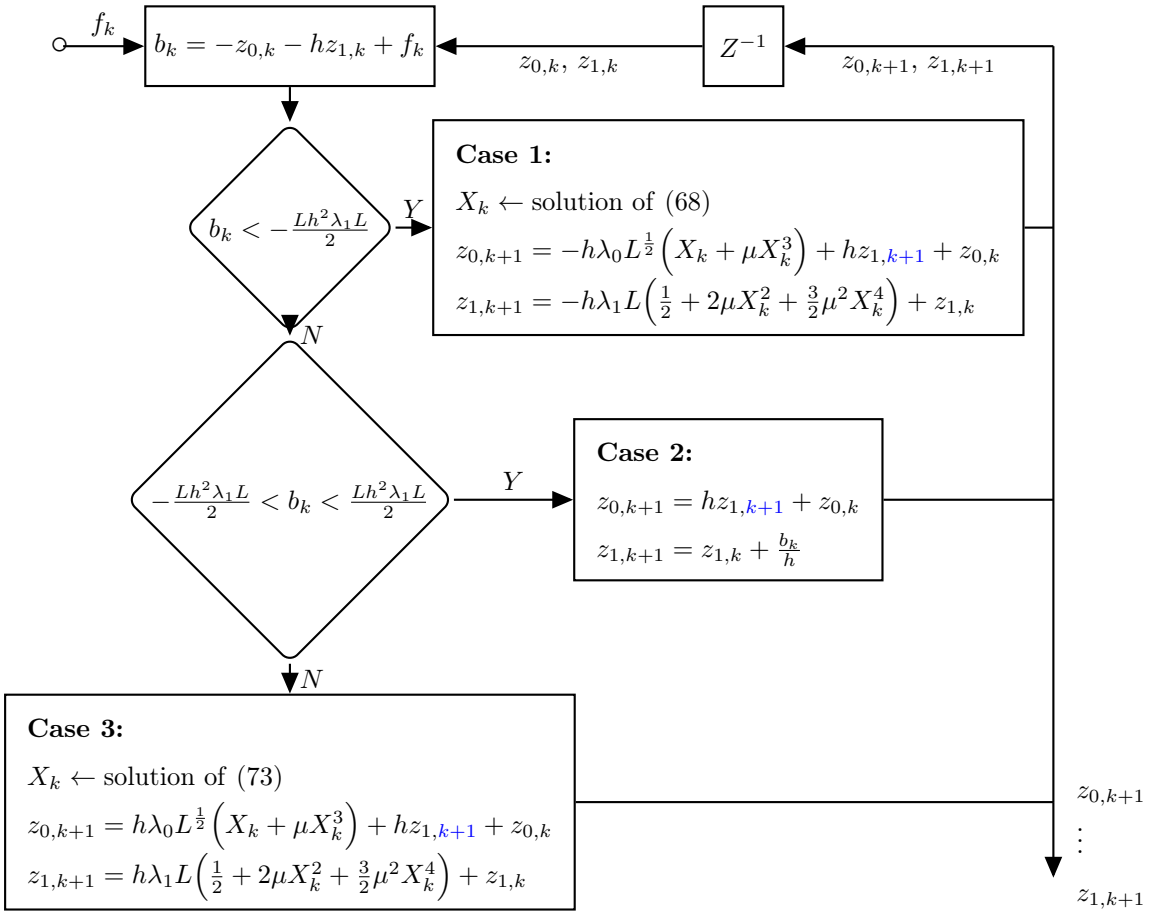

Figure 5: Flowchart of the I-URED. The block Z^{-1} indicates one-step delay.

The phase-portrait of the I-URED for $\lambda_0 = 2, \lambda_1 = 2$, $L = 1$, input signal $\sin(t)$, and different values of

μ, is shown in Fig. 6 for $h = 1$ms. It should be noted that for $\mu = 0$, the responses of I-URED and I-STD are the same. It can be seen that the estimation error of the input signal ($\sigma_{0,k}$) and the estimation error of the first-order differentiation ($\sigma_{1,k}$) converge to the origin for 20 randomly selected initial conditions. The initial values for $z_{0,k}$ and $z_{1,k}$ are selected within the interval $[-10, 10]$.

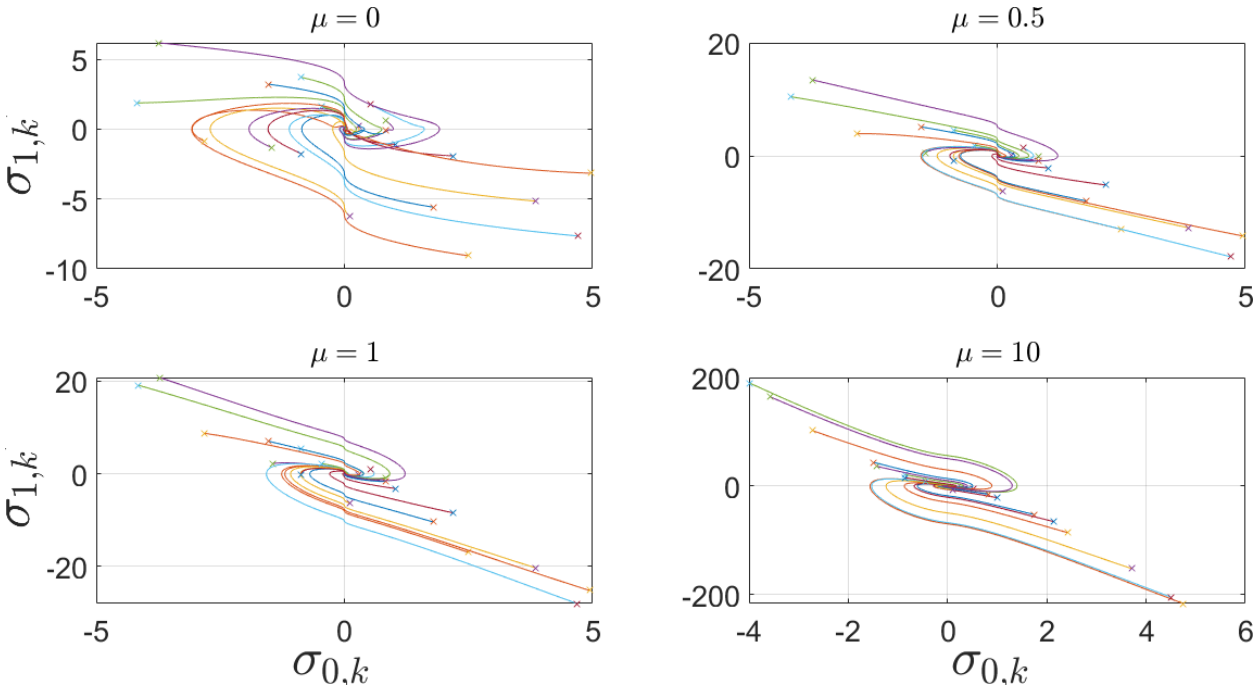

Figure 6: Phase-portrait of the I-URED. Input signal is $\sin(t)$. $\lambda_1 = 2, \lambda_2 = 2, h = 1$ms.

3.3.4 Implicit discretization of the arbitrary-order super-twisting differentiator

From (7), the implicit discretization of this differentiator gives:

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k+1} \rfloor^{\frac{n-i}{n+1}} + h z_{i+1,k+1} + z_{i,k}, \quad i = 0, \ldots, (n-1) & \text{(75a)} \\ z_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k+1}) + z_{n,k} \, . & \text{(75b)} \end{cases}$$

Substituting (75b) in (75a) leads to the following generalized equation:

$$\begin{cases} g(\sigma_{0,k+1}) \in -h^{n+1}\lambda_n L \operatorname{sgn}(\sigma_{0,k+1}) & \text{(76a)} \\ g(\sigma_{0,k+1}) = \sigma_{0,k+1} + \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} \lceil \sigma_{0,k+1} \rfloor^{\frac{n-l}{n+1}} \right) + b_k & \text{(76b)} \\ b_k = -\sum_{l=0}^{n} h^l z_{l,k} + f_k \, , \qquad \xi(\sigma_{0,k+1}) = g^{-1}(\sigma_{0,k+1}). & \text{(76c)} \end{cases}$$

Thus, in this case, the fundamental operator for the I-AO-STD is given by (77) and to be compared with (46).

$$x \mapsto \left(I_d + \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} \lceil \cdot \rfloor^{\frac{n-l}{n+1}}\right) + h^{n+1}\lambda_n L \operatorname{sgn}(\cdot)\right)^{-1}(-b_k) \tag{77}$$

This generalized equation will be solved as follows:

- **Case 1:** $b_k < -h^{n+1}\lambda_n L$

  Considering Fig. 2, $\sigma_{0,k+1} > 0$ is obtained for this case. Substituting $\operatorname{sgn}(\sigma_{0,k+1}) = 1$ in (76a) and (76b) yields

$$\begin{cases} g(\sigma_{0,k+1}) \in -h^{n+1}\lambda_n L & \text{(78a)} \\ g(\sigma_{0,k+1}) = \sigma_{0,k+1} + \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} (\sigma_{0,k+1})^{\frac{n-l}{n+1}}\right) + b_k. & \text{(78b)} \end{cases}$$

  Defining $x_k \triangleq \sigma_{0,k+1}^{\frac{1}{n+1}}$, and substituting it in (78) gives

$$x_k^{n+1} + \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} x_k^{n-l}\right) + b_k + h^{n+1}\lambda_n L = 0. \tag{79}$$

  Assuming that X_k is the solution of (79), $\sigma_{0,k+1}$ is given by $\sigma_{0,k+1} = X_k^{n+1}$.

- **Case 2:** $b_k \in [-h^{n+1}\lambda_n L, h^{n+1}\lambda_n L]$

  According to Fig. 2, $\sigma_{0,k+1} = 0$ is obtained for this case. Therefore, (76) gives

$$\begin{aligned} & b_k \in -h^{n+1}\lambda_n L \operatorname{sgn}(0) = -h^{n+1}\lambda_n L[-1,1] && \Leftrightarrow \\ & b_k = -h^{n+1}\lambda_n L\xi \qquad \text{for some } \xi \in [-1,1] && \Rightarrow \\ & \xi = \frac{b_k}{-h^{n+1}\lambda_n L}, \end{aligned} \tag{80}$$

  where ξ is called a selection of the set-valued signum function at zero. Substituting (80) into (75) yields

$$\begin{cases} z_{i,k+1} = h z_{i+1,k+1} + z_{i,k}, \quad i = 0, \ldots, (n-1) \\ z_{n,k+1} = z_{n,k} + \frac{b_k}{h^n} \end{cases} \tag{81}$$

- **Case 3:** $b_k > h^{n+1}\lambda_n L$

  In this case, $\sigma_{0,k+1} < 0$ is obtained from Fig. 2. Substituting $\operatorname{sgn}(\sigma_{0,k+1}) = -1$ yields

$$\begin{cases} g(\sigma_{0,k+1}) \in h^{n+1}\lambda_n L & \text{(82a)} \\ g(\sigma_{0,k+1}) = \sigma_{0,k+1} - \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} (-\sigma_{0,k+1})^{\frac{n-l}{n+1}}\right) + b_k. & \text{(82b)} \end{cases}$$

Defining $x_k \triangleq (-\sigma_{0,k+1})^{\frac{1}{n+1}}$, and substituting it in (82) gives

$$-x^{n+1} - \sum_{l=0}^{n-1} \left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}} x^{n-l}\right) + b_k - h^{n+1}\lambda_n L = 0. \tag{83}$$

The solution of (83) can be used to calculate $\sigma_{0,k+1}$, i.e., $\sigma_{0,k+1} = -x^{n+1}$.

In order to obtain the solution of the generalized equation (76), it is necessary to solve the polynomial equations (79) and (83) in an online manner for the cases 1 and 3, respectively. Note that for the case 2, (80) can be used to calculate $\text{sgn}(\sigma_{0,k+1})$ directly, and there is no need to solve any polynomial equation.

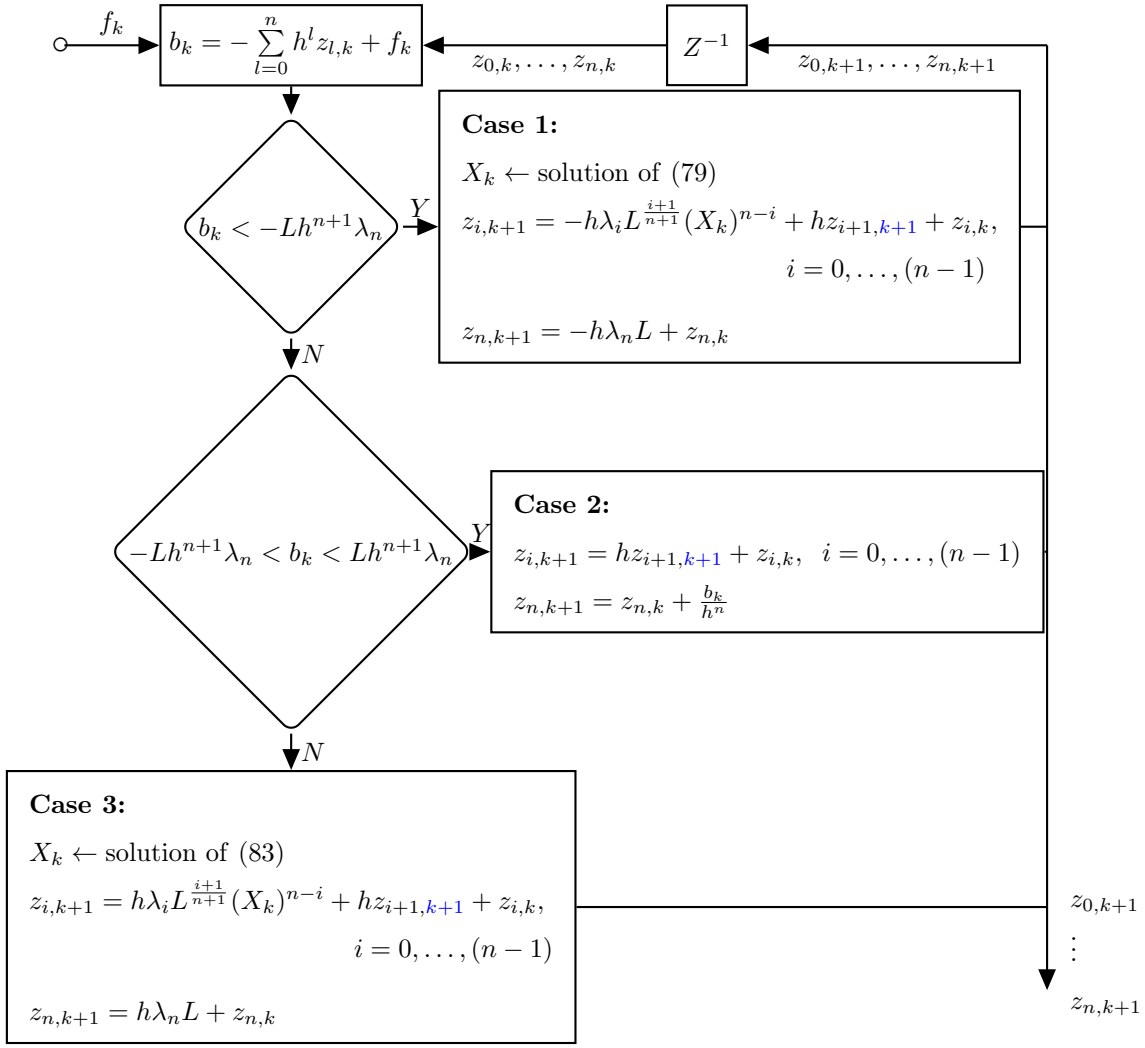

Figure 7: Flowchart of the I-AO-STD. The block Z^{-1} indicates one-step delay.

Remark 2 *According to the algorithms which are shown in Figs. 3, 5 and 7, it can be seen that the implicit discretization leads to a causal (non-anticipative) implementation since* $\text{sgn}(\sigma_{0,k+1})$ *can be always calculated at the time step* $k \geq 0$.

Remark 3 *Figs. 3, 5 and 7 show that the differentiation gains do not appear in the differentiation law for case 2. Therefore, in a noise-free case where $f(t) = f_0(t)$ for all $t \geq 0$, the implicit discretization is not sensitive to the differentiation gains as long as it remains in its sliding-phase (see Corollary 2). This is one of the main advantages of the implicit discretization, as noted for SMC [53, 57, 60]. However, in a noisy condition, there is not any guarantee that the implicit scheme keeps the sliding-phase (see Remark 7). Some simulations are provided in Section 5.7 to study the gain-insensitivity property of the implicit methods.*

Remark 4 *According to Fig. 2 it can be seen that the generalized equation has a unique solution. Furthermore, according to the Descartes' rule of sign [68] it is clear that the polynomials* (66)*,* (71)*,* (79) *and* (83) *also have unique solutions since there is only a single sign change among the coefficients.*

Remark 5 *In this work, a built-in MATLAB solver named* `roots` *has been used to solve the polynomial equations (see Section 5.8).*

Following [58], finite-time convergence of the I-AO-STD will be ensured based on the following procedures:

1. A crucial property to transport the Lyapunov stability properties from the continuous-time system to the discretized system is that the continuous-time Lyapunov function has convex level sets. Therefore, such a Lyapunov function will be introduced in Lemma 1.
2. In Lemma 2, it will be shown that the sliding surface of the discrete-time I-STD is invariant.
3. Asymptotic stability of the I-STD will be addressed in Lemma 3.
4. Finite-time convergence of the I-STD will be studied in Corollary 1.

Remark 6 *For the purpose of analysis, continuous-time STD* (6) *and AO-STD* (7) *will be rewritten in the following standard forms, respectively:*

$$\begin{cases} \dot{\sigma}_0 = -\lambda_0 L^{\frac{1}{2}} |\sigma_0|^{\frac{1}{2}} \operatorname{sgn}(\sigma_0) + \sigma_1 \\ \dot{\sigma}_1 \in -\lambda_1 L \operatorname{sgn}(\sigma_0) + \ddot{f}(t) \ , \end{cases} \tag{84}$$

$$\begin{cases} \dot{\sigma}_i(t) = -\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_0(t) \rfloor^{\frac{n-i}{n+1}} + \sigma_{i+1}(t), \ \ i = 0, \ldots, (n-1) \\ \dot{\sigma}_n(t) \in -\lambda_n L \operatorname{sgn}(\sigma_0(t)) + f^{(n+1)}(t), \end{cases} \tag{85}$$

where $\sigma_i(t) = z_i(t) - f^{(i)}(t), i = 0, \ldots, n$. For $f^{(n+1)}(t) = 0$, (84) *and* (85) *are in the homogeneous form.*

Lemma 1 *Considering $|\ddot{f}(t)| < L$, the system (84) admits a strict Lyapunov function $V(\cdot)$ for $L \geq 0$, $\lambda_0 > \sqrt{4L\sqrt{2}}$, and $L < \lambda_1 < \frac{\lambda_0^2}{2\sqrt{2}} - L$ with ellipsoidal level sets, and such that*

$$V \in C\big(\mathbb{R}^n, [0, +\infty]\big) \cap C^1\big(\mathbb{R}^n \backslash \{0\}, (0, +\infty)\big), \tag{86}$$

The proof is presented in Appendix B.

Definition 2 *The sliding surface of the I-AO-STD is defined as follows (for I-STD, one has $n = 1$):*

$$\Sigma_d \triangleq \{\sigma_k \in \mathbb{R}^n \mid \sigma_k = [\sigma_{0,k}, \dots, \sigma_{n,k}]^T = \mathbf{0}\}. \tag{87}$$

Lemma 2 *Let $f^{(n+1)}(t) = 0$ for all $t \geq 0$. Then the sliding surface of the implicit differentiators are invariant.*

See Appendix B for the proof.

Lemma 3 *Assume that $F : \mathbb{R}^n \to \mathbb{R}^n$ is an upper semi-continuous transformation map such that $F(\sigma)$ is nonempty convex and compact for all $\sigma \in \mathbb{R}^n$. Let the differential inclusion $\dot{\sigma} \in F(\sigma(t))$ have a unique globally asymptotically stable equilibrium point at the origin. If its strict Lyapunov function (86) has convex level sets, then any sequence $\{\sigma_{0,k}\}_{k=0,\dots,\infty}$ generated by the difference inclusion $\sigma_{0,k+1} \in hF(\sigma_{0,k+1}) + \sigma_{0,k}$ converges to zero as $k \to \infty$, for any bounded initial data.*

The proof is given in Appendix B.

Corollary 1 *Assume that Lemma 3's results hold here. Then the sliding phase is achieved ($\Sigma_d = \mathbf{0}$) in a finite number of steps if $f^{(n+1)}(t) = 0$ for all $t \geq 0$.*

The proof is presented in Appendix B. Notice that Lemma 3 holds only in case $n = 1$, using Lemma 1.

Corollary 2 *Let $f^{(n+1)}(t) = 0$, then a sufficient condition for keeping the sliding-phase of the I-AO-STD in Fig. 7, is $\big|\sum_{i=1}^{n-1}\big((n-i)h^i f_k^{(i)}\big) + f_{k+1} - f_k\big| < Lh^{n+1}\lambda_n$, for all $h \geq 0$.*

Proof. From (76c), one has

$$b_{k+1} = -\sum_{l=0}^{n} h^l z_{l,k+1} + f_{k+1}. \tag{88}$$

Substituting (81), which holds for $f^{(n+1)}(t) = 0$ and Case 2, into (88), and keeping in mind that $b_k = -\sum_{l=0}^{n} h^l z_{l,k} + f_k$ one has

$$\big|\sum_{i=0}^{n-1}\big((n-i)h^i z_{i,k}\big) + f_{k+1} - (n+1)f_k\big| < Lh^{n+1}\lambda_n. \tag{89}$$

Assuming that the sliding-phase was achieved at the time step k, $z_{i,k} = f_k^{(i)}$ holds. Hence, (89) gives,

$$\big|\sum_{i=1}^{n-1}\big((n-i)h^i f_k^{(i)}\big) + f_{k+1} - f_k\big| < Lh^{n+1}\lambda_n. \tag{90}$$

which is the sufficient condition for keeping the sliding-phase.

Remark 7 *If the parameters are designed according to* (90)*, the I-AO-STD will track the exact differentiation of $f_k = f_{0,k} + n_k$ including the high frequency noise (n_k) during the sliding-phase, where $f_{0,k}$ is the signal which is corrupted. Therefore, from an engineering point of view, the parameters may be designed as follows to obtain a trade-off between the exactness and the robustness:*

$$\begin{cases} \big|\sum_{i=1}^{n-1}\big((n-i)h^i f_{0,k}^{(i)}\big) + f_{0,k+1} - f_{0,k}\big| < Lh^{n+1}\lambda_n & \text{(91a)} \\ \big|\sum_{i=1}^{n-1}\big((n-i)h^i n_k^{(i)}\big) + n_{k+1} - n_k\big| \gg Lh^{n+1}\lambda_n. & \text{(91b)} \end{cases}$$

As it will be seen in Section 5, $\big|\sum_{i=1}^{n-1}\big((n-i)h^i n_k^{(i)}\big) + n_{k+1} - n_k\big| \gg Lh^{n+1}\lambda_n$ may not hold because of the overlap between the frequency components of the signal and noise. In this case, it is not possible to satisfy both (91a) and (91b) at the same time, and it might be necessary to select a smaller L which does not satisfy (91a). It will decrease the effects of the input noise (decrease the chattering caused by a high-frequency noise). However, it also cancels the exactness of the I-STD for $f_{0,k}$. For instance, for $f_k = \sin(hk)$ and $h = 50$ms, $|f_{k+1} - f_k| < 0.05$ holds. Thus, for $\lambda = 1.1$, $\frac{0.05}{h^2\lambda_1} = 18.2 < L$ ensures the exactness of the I-STD on $f_{0,k}$. However, as it will be seen, the optimization procedure (see Section 4) always selects much smaller L, i.e., $L \approx 1$ to cancel the exactness of the I-STD for the noise (i.e., to improve the robustness). Of course without considering the noise, the optimization procedure always selects $L \geq 18.2$.

Remark 8 *Assuming $n = 1$, and dividing* (91a) *by $h > 0$ yields:*

$$\Big|\frac{f_{0,k+1} - f_{0,k}}{h}\Big| < Lh\lambda_1 \quad \Rightarrow \quad |f_{0,k}^{(1)}| < Lh\lambda_1. \tag{92}$$

Furthermore, since it is assumed that for the I-STD the second derivative of $f(\cdot)$ is uniformly bounded, the first derivative is Lipschitz continuous as follows:

$$|f_{0,k}^{(2)}| < L \quad \Rightarrow \quad |f_{0,k}^{(1)}| < Lh. \tag{93}$$

Considering (92) *and* (93)*, a sufficient condition for keeping the sliding-phase for the I-STD (n=1) is:*

$$\lambda_1 > 1. \tag{94}$$

This is compatible with Table 2 in Section 4, where $\lambda_n = 1.1$ is always considered (n is the order of the differentiator). In this work, $\lambda_1 = 1.1$ will be considered for the simulations. Moreover, the parameter L will be selected based on an optimization procedure to satisfy (91).

Lemma 4 *Let $\ddot{f}(t) = 0$ for all $t \geq 0$ and $|f_k - f_{k+1}| < Lh^2\lambda_1$. Then, the following inequality holds.*

$$|z_{1,k} - f_k^{(1)}| < Lh\lambda_1 \tag{95}$$

Proof. Considering (50) and (95), one has

$$|z_{1,k+1} - f_{k+1}^{(1)}| < Lh\lambda_1 \quad \Longleftrightarrow \tag{96a}$$

$$\Big| - \frac{z_{0,k} - f_k}{h} - \frac{f_{k+1} - f_k}{h}\Big| < Lh\lambda_1 \quad \Longleftrightarrow \tag{96b}$$

$$|f_k - f_{k+1}| < Lh^2\lambda_1, \tag{96c}$$

where $|f_k - f_{k+1}| < Lh^2\lambda_1$ is just the condition for tuning the parameters which was obtained in Corollary 2 and always holds. ■

Numerical experiments: The inequality (95) has been investigated using numerical simulations as shown in Fig. 8. The responses of the E-STD and I-STD for four different inputs, i.e., $f(t) = \sin(t)$, $f(t) = 5t^2 + 3$, $f(t) = 2\sin(t) + 3\cos(2t) + 4t$ and $f(t) = t^3$ are shown in Fig. 8 (A) to (D), respectively. It can be seen that the absolute maximum output error, i.e., L_∞ norm of the output error for the implicit method is always less than the worst case band which is presented in Fig. 8 in the cases where the second-order derivation is bounded. The worst case band is provided by the equation $|z_{1,k} - f_k^{(1)}| = Lh\lambda_1$. On the other hand, the output error of the explicit one does not satisfy (95). Note that for the input $f(t) = t^3$, the second-order derivation is not bounded. As the result, neither implicit nor explicit schemes satisfy (95) as it was expected (see Fig. 8 (D)).

It should be noted that, in the literature [33] (see Theorem 1), the inequality $|z_{1,k} - f_k^{(1)}| < \mu h$ is obtained for the E-STD with an unknown parameter μ. In this study it is shown that, for $\mu = L\lambda_n$, the inequality (95) always holds for the implicit discretization while the explicit one may not satisfy that inequality as shown in Fig. 8.

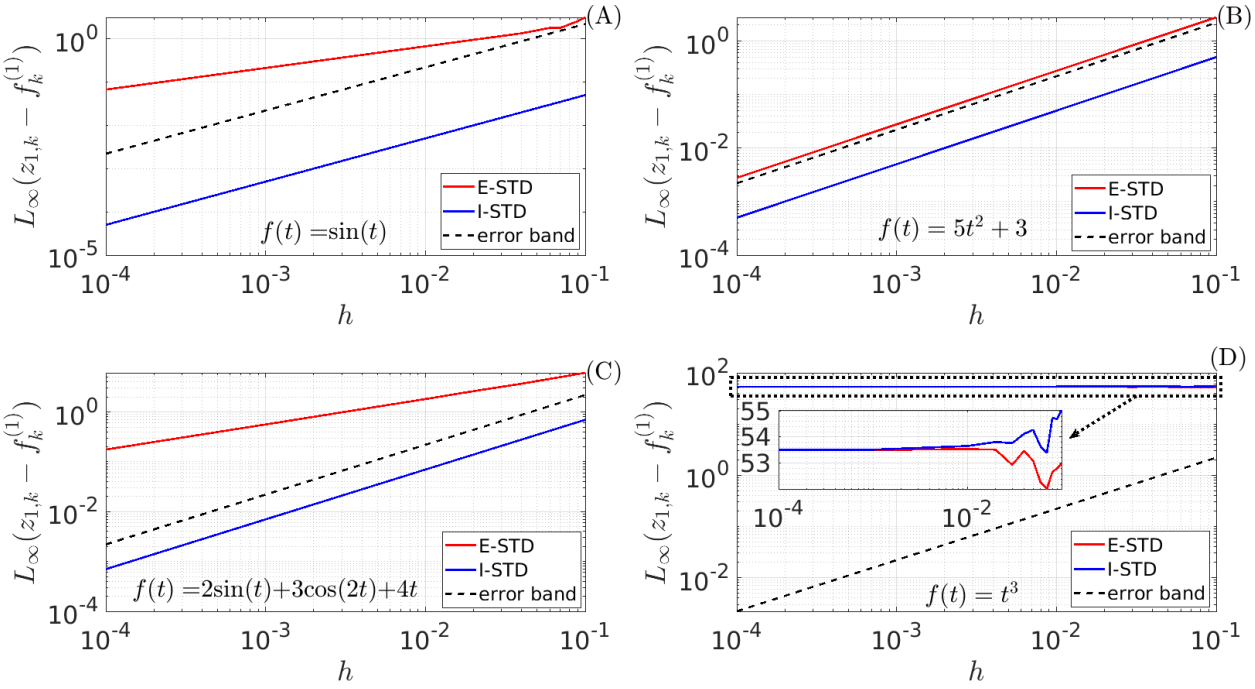

Figure 8: Investigation of the (95) for I-STD and E-STD. $L = 20$, $\lambda_1 = 1.1, n = 1, i = 1$

Considering the noise, according to (91b), we may need to select a smaller L which does not satisfy (91a). In this case, the I-STD may not reach the sliding surface (see Corollary 2) and therefore the inequality which is obtained in (95) may not hold.

3.3.5 Implicit discretization of HDD

The explicit discretization of HDD [33] is given in (24). The implicit discretization of this differentiator can be obtained as follows:

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k+1} \rfloor^{\frac{n-i}{n+1}} + \sum_{j=1}^{n-i} \frac{h^j}{j!} z_{j+1,k+1} + z_{i,k}, \quad i = 0, \ldots, (n-1) & \text{(97a)} \\ z_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k+1}) + z_{n,k}. & \text{(97b)} \end{cases}$$

Substituting (97a) in (97b) leads to the following generalized equation (note that the differences between the generalized equation of the I-AO-STD (76) and I-HDD (97) are indicated with red color. It should also be noted that the coefficients $m_i, i = 0, \ldots, n$ are naturally appear in the generalized equation for any given order n):

$$\begin{cases} g(\sigma_{0,k+1}) \in -h^{n+1}\lambda_n L m_n \operatorname{sgn}(\sigma_{0,k+1}) & \text{(98a)} \\ g(\sigma_{0,k+1}) = \sigma_{0,k+1} + \sum_{l=0}^{n-1} \left(m_l h^{l+1} \lambda_l L^{\frac{l+1}{n+1}} \lceil \sigma_{0,k+1} \rfloor^{\frac{n-l}{n+1}} \right) + b_k & \text{(98b)} \\ b_k = -\sum_{l=0}^{n} m_l h^l z_{l,k} + f_k \,, \qquad \xi(\sigma_{0,k+1}) = g^{-1}(\sigma_{0,k+1}). & \text{(98c)} \end{cases}$$

where

$$m_0 = 1, \quad m_1 = 1, \quad m_2 = \frac{3}{2}, \quad m_3 = \frac{13}{6}, \quad m_4 = \frac{25}{8}, \quad m_5 = \frac{541}{120}, \quad m_6 = \frac{1561}{240}. \tag{99}$$

In this case, the fundamental operator for the I-HDD is given by (100) and to be compared with (77).

$$x \mapsto \left(I_d + \sum_{l=0}^{n-1} \left(m_l h^{l+1} \lambda_l L^{\frac{l+1}{n+1}} \lceil \cdot \rfloor^{\frac{n-l}{n+1}} \right) + h^{n+1}\lambda_n L m_n \operatorname{sgn}(\cdot) \right)^{-1} (-b_k) \tag{100}$$

Comparing (76) and (98), it can be seen that the algorithm for the implicit discretization of HDD can be obtained in the same manner as in Section 3.3.4. For the sake of simplicity, flowchart of the implicit discretization of the HDD is provided in Fig. 9. Similar to other implicit schemes, the following polynomial equations have to be solved for the I-HDD to advance the algorithm from step k to step $k+1$ (the polynomial is obtained for the third-order implicit HDD):

$$\begin{cases} x_k^4 + a_3 x_k^3 + a_2 x_k^2 + a_1 x_k + b_k + a_0 = 0 & \text{(101a)} \\ a_3 = m_0 h \lambda_0 L^{\frac{1}{4}} & \text{(101b)} \\ a_2 = m_1 h^2 \lambda_1 L^{\frac{2}{4}} & \text{(101c)} \\ a_1 = m_2 \frac{3}{2} h^3 \lambda_2 L^{\frac{3}{4}} & \text{(101d)} \\ a_0 = m_3 \frac{13}{6} h^4 \lambda_3 L & \text{(101e)} \\ \sigma_{0,k+1} = X_k^4, & \text{(101f)} \end{cases}$$

and:

$$\begin{cases} -x_k^4 - a_3 x_k^3 - a_2 x_k^2 - a_1 x_k - (a_0 - b_k) = 0 & \text{(102a)} \\ a_3 = m_0 h \lambda_0 L^{\frac{1}{4}} & \text{(102b)} \\ a_2 = m_1 h^2 \lambda_1 L^{\frac{2}{4}} & \text{(102c)} \\ a_1 = m_2 \dfrac{3}{2} h^3 \lambda_2 L^{\frac{3}{4}} & \text{(102d)} \\ a_0 = m_3 \dfrac{13}{6} h^4 \lambda_3 L & \text{(102e)} \\ \sigma_{0,k+1} = X_k^4, & \text{(102f)} \end{cases}$$

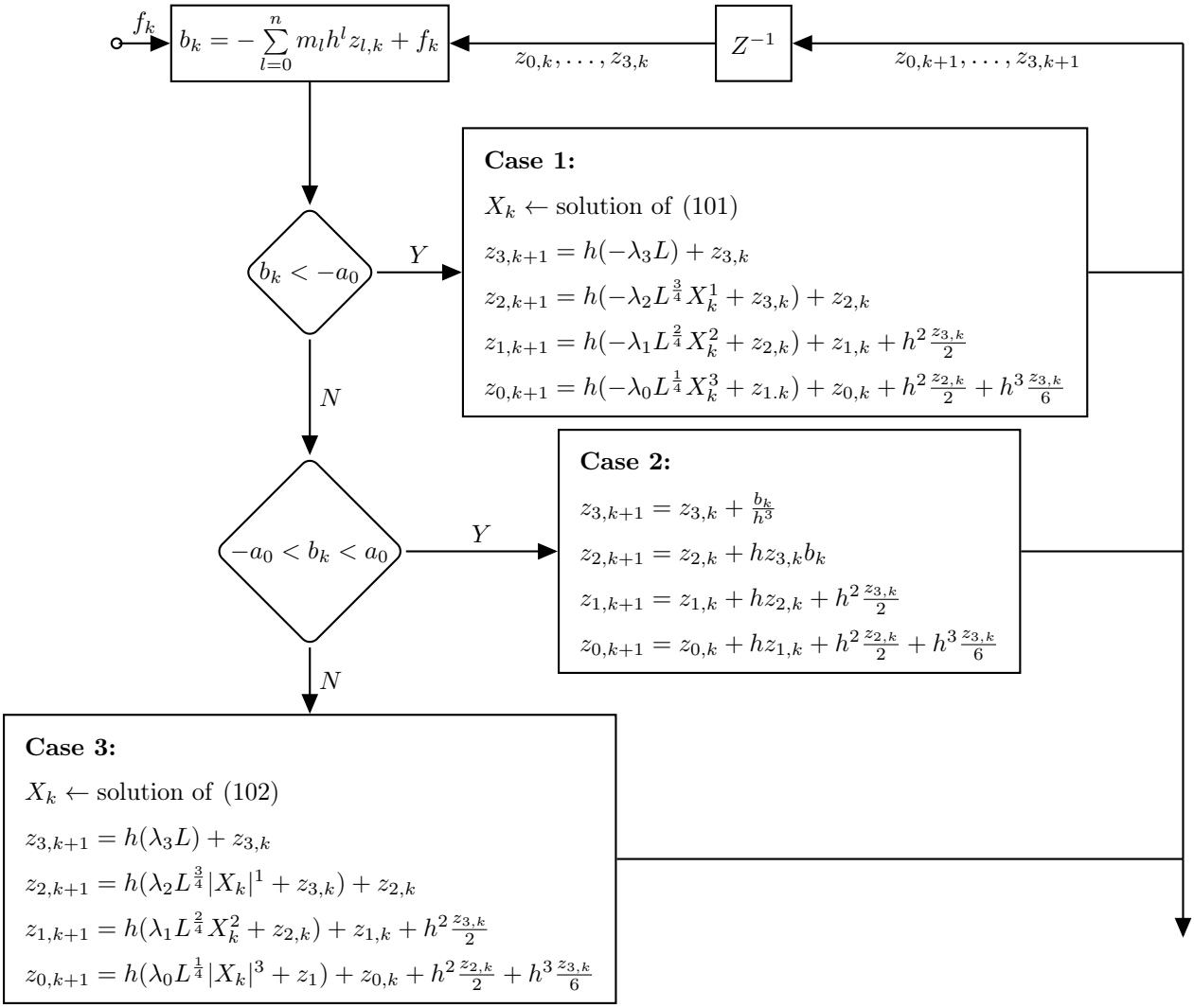

Figure 9: Flowchart of the third-order I-HDD. The block Z^{-1} indicates one-step delay.

3.3.6 Implicit discretization of GHDD

The implicit discretization of the third-order GHDD (33) gives:

$$\begin{cases} z_{0,k+1} = z_{0,k} + hz_{1,k+1} + \dfrac{h^2}{2} z_{2,k+1} + \dfrac{h^3}{6} z_{3,k+1} + h\psi_{0,k+1} & \text{(103a)} \\ z_{1,k+1} = z_{1,k} + hz_{2,k+1} + \dfrac{h^2}{2} z_{3,k+1} + h\psi_{1,k+1} + \alpha_{12}h^2\psi_{2,k+1} + \alpha_{13}h^3\psi_{3,k+1} & \text{(103b)} \\ z_{2,k+1} = z_{2,k} + hz_{3,k+1} + h\psi_{2,k+1} + \alpha_{23}h^2\psi_{3,k+1} & \text{(103c)} \\ z_{3,k+1} \in z_{3,k} + h\psi_{3,k+1}, & \text{(103d)} \end{cases}$$

where $\alpha_{12} = -\frac{1}{2}, \alpha_{23} = -1$ and $\alpha_{13} = \frac{1}{3}$ and,

$$\begin{aligned} \psi_{0,k+1}(\sigma_{0,k+1}) &= -\lambda_0 L^{\frac{1}{4}} \lceil \sigma_{0,k+1} \rceil^{\frac{3}{4}} \\ \psi_{1,k+1}(\sigma_{0,k+1}) &= -\lambda_1 L^{\frac{1}{2}} \lceil \sigma_{0,k+1} \rceil^{\frac{1}{2}} \\ \psi_{2,k+1}(\sigma_{0,k+1}) &= -\lambda_2 L^{\frac{3}{4}} \lceil \sigma_{0,k+1} \rceil^{\frac{1}{4}} \\ \psi_{3,k+1}(\sigma_{0,k+1}) &= -\lambda_3 L \operatorname{sgn}(\sigma_{0,k+1}). \end{aligned} \tag{104}$$

Substituting (103b) to (103d) into (103a) leads to the following generalized equation:

$$\begin{aligned} &z_{0,k+1} \in z_{0,k} + h\psi_{0,k+1} + h\Big(z_{1,k} + hz_{2,k+1} + \tfrac{h^2}{2} z_{3,k+1} + h\psi_{1,k+1} + \cancel{\alpha_{12}h^2\psi_{2,k+1}} + \cancel{\alpha_{13}h^3\psi_{3,k+1}}\Big) \\ &+ \tfrac{h^2}{2!}\Big(z_{2,k} + hz_{3,k+1} + \cancel{h\psi_{2,k+1}} + \cancel{\alpha_{23}h^2\psi_{3,k+1}}\Big) \\ &+ \tfrac{h^3}{3!}\Big(z_{3,k} + \cancel{h\psi_{3,k+1}}\Big). \end{aligned} \tag{105}$$

Hence, (105) gives:

$$z_{0,k+1} \in z_{0,k} + hz_{1,k} + \tfrac{3h^2}{2} z_{2,k} + \tfrac{13h^3}{6} z_{3,k} + h\psi_{0,k+1} + h^2\psi_{1,k+1} + h^3\psi_{2,k+1} + h^4\psi_{3,k+1}. \tag{106}$$

Thus, in general, the generalized equation of the I-GHDD is given by,

$$z_{0,k+1} - f_k \in \sum_{i=0}^{n} (m_i h^i z_{i,k}) + \sum_{i=0}^{n} h^{i+1}\psi_{i,k+1} - f_k, \tag{107}$$

where the quantities m_i are provided in (99). The generalized equation can thus be rewritten as follows:

$$\begin{cases} g(\sigma_{0,k+1}) \in -h^{n+1} L\lambda_n \operatorname{sgn}(\sigma_{0,k+1}) & \text{(108a)} \\ g(\sigma_{0,k+1}) = \sigma_{0,k+1} - \displaystyle\sum_{i=0}^{n-1} \left(h^{i+1}\psi_{i,k+1}\right) + b_k & \text{(108b)} \\ b_k = -\displaystyle\sum_{i=0}^{n} \left(m_i h^i z_{i,n}\right) + f_k \ , \qquad \xi(\sigma_{0,k+1}) = g^{-1}(\sigma_{0,k+1}). & \text{(108c)} \end{cases}$$

In this case, the fundamental operator for the I-GHDD is given by (109).

$$x \mapsto \left(I_d - \sum_{i=0}^{n-1} \left(h^{i+1}\psi_{i,k+1}(\cdot)\right) + h^{n+1} L\lambda_n \operatorname{sgn}(\cdot)\right)^{-1}(-b_k) \tag{109}$$

This generalized equation can be solved similarly to the I-AO-STD which was proposed in Section 3.3.4. The flowchart of the third-order I-GHDD is shown in Fig. 10.

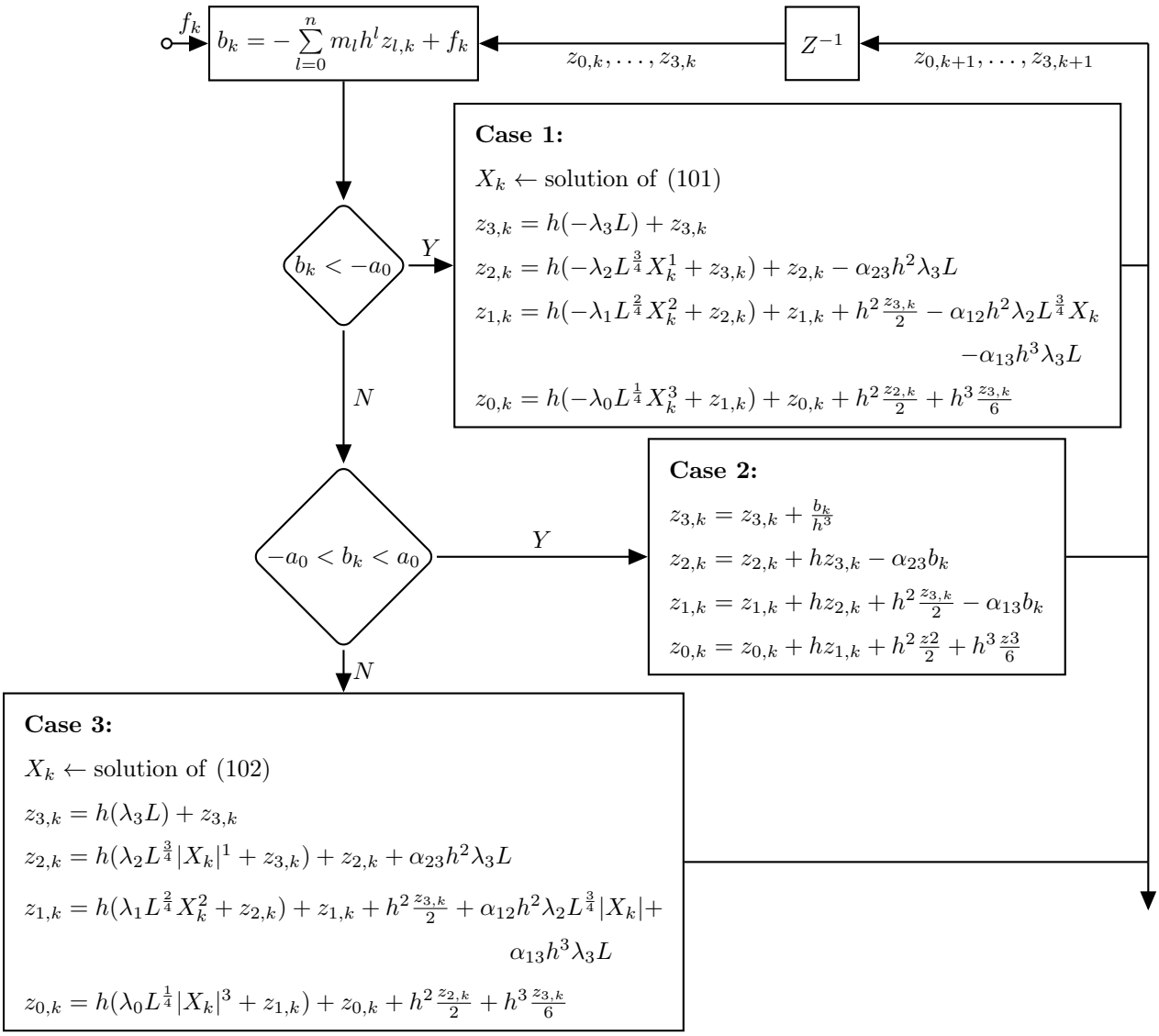

Figure 10: Flowchart of the third-order I-GHDD. The block Z^{-1} indicates one-step delay.

3.3.7 Implicit discretization of VGED

Considering the explicit scheme in (35), the implicit discretization of VGED reads as:

$$\begin{cases} z_{0,k+1} = -h\lambda_0 \mu |\sigma_{0,k+1}|^{\alpha_k} \operatorname{sgn}(\sigma_{0,k+1}) + h z_{1,k+1} + z_{0,k} & \text{(110a)} \\ z_{1,k+1} = -h\lambda_1 \alpha_k \mu^2 |\sigma_{0,k+1}|^{2\alpha_k - 1} \operatorname{sgn}(\sigma_{0,k+1}) + z_{1,k} & \text{(110b)} \\ \gamma_{k+1} = -h\tau\gamma_k + h\tau |f_{f,k}| + \gamma_k & \text{(110c)} \\ \alpha_k = \frac{1}{2}\left(1 + \frac{\gamma_k^q}{\gamma_k^q + \epsilon}\right). & \text{(110d)} \end{cases}$$

From (110), it can be seen that, unlike STD, the VGED has a continuous right-hand side. Hence, considering the full-implicit scheme, there will not be any set-valued part to be calculated during the selection procedure.

Furthermore, the set-valued parts may appear in the semi-implicit schemes. It should be noted that implicit discretization may also be able to provide useful characteristics where the set-valued parts are absent.

3.3.8 Implicit discretization of the SHMD

Investigations reveal that the explicit (forward) discretization of the SHMD shows a significant amount of digital chattering since the discontinuous signum function appears in all the inclusions of (5) [16, 17]. Explicit discretization of (5) is considered in Fig. 11 to show the output chattering of the explicit SHMD.

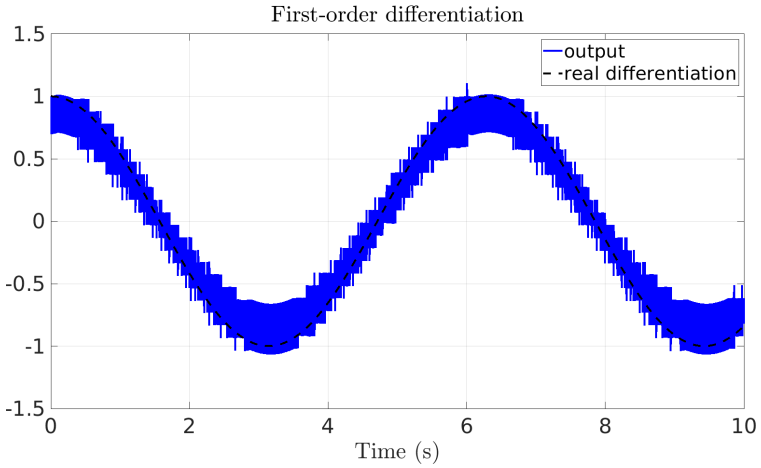

Figure 11: Output of the SHMD (5) discretized using explicit method for the input $f(t) = \sin(t)$ ($n = 1, h =$ 1ms).

As can be seen from Fig. 11, explicit discretization of the SHMD contains too much chattering even for the small sampling time $h = 1$ms. Hence, the explicit discretization of this differentiator is ignored throughout this study. The implicit discretization of the first- and arbitrary-order SHMD has been previously studied in [16, 17], respectively. However, another implicit representation of the SHMD is developed in the sequel to provide a unified representation.

Considering the SHMD in the continuous-time setting (5) with $\Psi(\cdot) = \operatorname{sgn}(\cdot)$, the implicit discretization gives

$$\begin{cases} z_{i,k+1} \in -h\alpha_i \operatorname{sgn}(\sigma_{0,k+1}) - h\kappa_i\sigma_{0,k+1} + hz_{i+1,k+1} + z_{i,k}, \qquad i = 0,\dots,n-1 & \text{(111a)} \\ z_{n,k+1} \in -h\alpha_n \operatorname{sgn}(\sigma_{0,k+1}) - h\kappa_n\sigma_{0,k+1} + z_{n,k}. & \text{(111b)} \end{cases}$$

Substituting (111b) into (111a) yields,

$$\begin{cases} a\sigma_{0,k+1} + b_k \in -c\operatorname{sgn}(\sigma_{0,k+1}) \\ b_k = -\sum_{l=0}^{n}(h^l z_{l,k}) + f_k \\ c = \sum_{l=0}^{n}(\alpha_l h^{l+1}) \\ a = 1 + \sum_{l=0}^{n}(\kappa_l h^{l+1}) \end{cases} \tag{112}$$

In this case, the fundamental operator for the SHMD (I-FDFF and I-AO-FDFF) is given by (113).

$$x \mapsto \Big(aI_d + c\,\mathrm{sgn}(\cdot)\Big)^{-1}(-b_k) \tag{113}$$

Since (112) represents a first-order GE, Fig. 15 with $n = 1$ can be used to solve this GE according to the following conditions:

- **Case 1:** $b_k < -c$

  In this case, $\sigma_{0,k+1} > 0$ is obtained from Fig. 15. Hence one has

$$\sigma_{0,k+1} = -\frac{c + b_k}{a} \tag{114}$$

- **Case 2:** $b_k \in [-c, c]$

  According to Fig. 15, $\sigma_{0,k+1} = 0$ is obtained for this case. Therefore, (112) gives

$$\begin{aligned} & b_k \in -c\,\mathrm{sgn}(0) = -c[-1, 1] && \Leftrightarrow \\ & b_k \in -c\xi \qquad \text{for some } \xi \in [-1, 1] && \Rightarrow \\ & \xi = -\tfrac{b_k}{c} \end{aligned} \tag{115}$$

  where ξ is called a selection of the set-valued signum function at zero.

- **Case 3:** $b_k > c$

  In this case, $\sigma_{0,k+1} < 0$ is obtained from Fig. 15. Hence one has

$$\sigma_{0,k+1} = \frac{c - b_k}{a} \tag{116}$$

The flowchart of the implicit SHMD is provided in Fig. 12. According to [16], for $n = 1$, implicit SHMD is termed I-FDFF. Furthermore, for $n > 1$, this differentiator is called I-AO-FDFF with this difference that an extra filtration is utilized as follows instead of directly using $z_i, i = 0, \ldots, n$ as outputs. More clearly, $w_i, i = 0, \ldots, n$ indicates the estimation of the i-th order differentiation for the I-AO-FDFF as follows:

$$w_{i,k} = hF\,\mathrm{sat}\left(\frac{h\epsilon z_{i,k} + w_{i,k-1}}{hF(h\epsilon + 1)} - \frac{w_{i,k-1}}{hF}\right), \tag{117}$$

where F and ϵ are the filtering parameters.

For the I-FDFF the parameters are designed as follows [16]:

$$\alpha_1 = \gamma, \quad \alpha_2 = \gamma\omega_s, \quad k_1 = 2\omega_f, \quad k_2 = \omega_f^2, \tag{118}$$

where $\gamma, \omega_s, \omega_f > 0$ are some parameters to be tuned. On the other hand, for a third-order I-AO-FDFF, the parameters are designed as follows [17]:

$$\kappa_0 = 2.613\omega_f,\ \kappa_1 = 3.414\omega_f^2,\ \kappa_2 = 2.613\omega_f^3,\ \kappa_3 = \omega_f^4,\ \alpha_1 = 2\alpha_1\omega_s,\ \alpha_2 = 2\alpha_1\omega_s^2,\ \alpha_3 = \alpha_1\omega_s^3, \tag{119}$$

where ω_s and ω_f are cut-off frequencies. These parameters as well as α_1 will be obtained using an optimization algorithm. Note that another example for the set-valued sign function is provided in [16, 17] as (120), which is not set-valued but a kind of saturation. Here, it is reminded that all the forthcoming simulations will be conducted based on the modified $\Psi(\cdot)$ which is presented in (120).

$$\Psi(x) = \begin{cases} x & |x| \leq 1 \\ (\rho|x|^{\frac{1}{p}}\rho + 1)\,\mathrm{sgn}(x) & |x| > 1 \end{cases} \tag{120}$$

where $\rho > 1$.

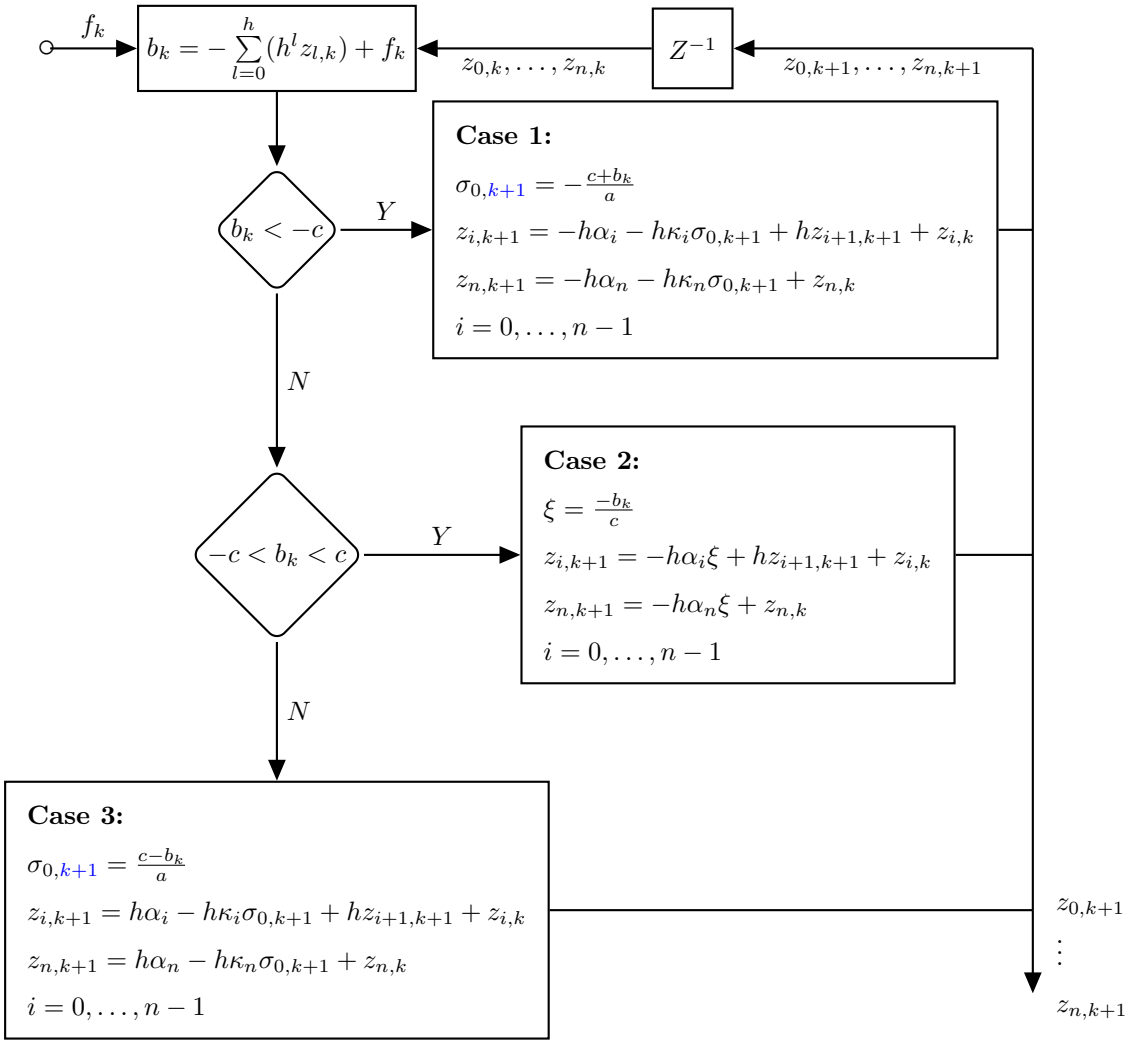

Figure 12: Flowchart of the implicit SHMD . The block Z^{-1} indicates one-step delay.

3.4 Semi-implicit discretization of the exact differentiators

So far, the full-implicit discretization method was deeply analyzed in Section 3.3. According to the literature (see [58]), full-implicit discretization may be necessary to ensure the convergence of the discrete-time exact differentiators to their continuous-time form. However, full-implicit discretization may lead to high-order

polynomial equations that require extra computations. To resolve this issue, semi-implicit methods are introduced next, to provide a simpler implementation as well as achieving a few useful properties of the implicit methods. Several semi-implicit discretization schemes are presented below. Note that explicit and implicit terms are shown in red and blue colors, respectively.

To be more precise, suppose that we deal with the ODE: $\dot{x}(t) = f_1\big(x(t)\big) + f_2\big(x(t)\big)$. Different time-discretization schemes can be formulated as follows:

$$\begin{aligned} &\text{Explicit (forward):} && x_{k+1} = hf_{1,k} + hf_{2,k} + x_k \\ &\text{Implicit (backward):} && x_{k+1} = hf_{1,k+1} + hf_{2,k+1} + x_k \\ &\text{Semi-implicit:} && \begin{cases} x_{k+1} = hf_{1,k+1} + hf_{2,k} + x_k \\ x_{k+1} = hf_{1,k} + hf_{2,k+1} + x_k. \end{cases} \end{aligned} \tag{121}$$

3.4.1 The first semi-implicit scheme

The first semi-implicit method for AO-STD leaves the set-valued part explicit, and is designed as follows:

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n-i}{n+1}} + hz_{i+1,k+1} + z_{i,k}, \quad i = 0, \dots, (n-1) & \text{(122a)} \\ z_{n,k+1} = -h\lambda_n L \operatorname{sgn}(\sigma_{0,k}) + z_{n,k} \ , & \text{(122b)} \end{cases}$$

where $\sigma_{0,k+1} = z_{0,k+1} - f_k$. From this equation, it can be seen that this semi-implicit method will not lead to a generalized equation since the discontinuous part is explicitly discretized. In fact, because of the triangular form of (122), it can be solved easily. Based on this semi-implicit discretization method, STD can also be implemented according to (122) as follows:

$$\begin{cases} z_{0,k+1} = -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} + hz_{1,k+1} + z_{0,k} & \text{(123a)} \\ z_{1,k+1} = -h\lambda_1 L \operatorname{sgn}(\sigma_{0,k}) + z_{1,k}. & \text{(123b)} \end{cases}$$

While it is shown that full-implicit method is insensitive to the gains during the sliding phase (see *Case 2* in Fig. 7), it can be seen that all parameters (L and λ_i) always appear in (122) and (123). As a result, this scheme is sensitive to the parameters even in the sliding-phase (see also Section 5.7). For both (122) and (123) the equality replaces $\in$ because of the explicit discretization. Indeed in this case, one is led to considering the single-valued signum function [53].

3.4.2 The second semi-implicit scheme

The second semi-implicit method for AO-STD is shown in (124). Compared to (122) and (123), it adds implicit terms in the set-valued part of the differentiator. But the single-valued terms are left explicit. It reads as:

$$\begin{cases} z_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k} \rfloor^{\frac{n-i}{n+1}} + hz_{i+1,k+1} + z_{i,k}, \quad i = 0, \dots, (n-1) & \text{(124a)} \\ z_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k+1}) + z_{n,k} \ . & \text{(124b)} \end{cases}$$

The corresponding STD is:

$$\begin{cases} z_{0,k+1} = -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} + h z_{1,k+1} + z_{0,k} & \text{(125a)} \\ z_{1,k+1} \in -h\lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) + z_{1,k}. & \text{(125b)} \end{cases}$$

Considering (125), the generalized equation to be solved at each step is as follows:

$$z_{0,k+1} \in -h\lambda_0 L^{\frac{1}{2}} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} - h^2 \lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) + h z_{1,k} + z_{0,k}. \tag{126}$$

The GE in (126) can be rewritten as:

$$\begin{cases} \sigma_{0,k+1} + b_k \in -h^2 \lambda_1 L \operatorname{sgn}(\sigma_{0,k+1}) & \text{(127a)} \\ b_k = h\lambda_0 \sqrt{L} \lceil \sigma_{0,k} \rfloor^{\frac{1}{2}} - h z_{1,k} - z_{0,k} + f_k. & \text{(127b)} \end{cases}$$

In this case, the fundamental operator of this semi-implicit scheme is given by (128).

$$x \mapsto \Big(I_d + h^2 \lambda_1 L \operatorname{sgn}(\cdot) \Big)^{-1} (-b_k) \tag{128}$$

Compared with (44) one sees that the GE in (127) is simplified, with a modified term b_k (compare (127b) and (76)). Thus, (127) is a classical GE, already met in SMC [54] and contact mechanics. Let us provide details. This semi-implicit STD can be implemented as follows:

- **Case 1**: $b_k < -h^2 \lambda_1 L$

  From Fig. 2, $\operatorname{sgn}(\sigma_{0,k+1}) = 1$ is obtained. Thus,

$$\sigma_{0,k+1} = -h^2 \lambda_1 L - b_k. \tag{129}$$

- **Case 2**: $-h^2 \lambda_1 L \leq b_k \leq h^2 \lambda_1 L$

  The selection of the set-valued signum function is as follows:

$$\begin{aligned} & b_k \in -h^2 \lambda_1 L \operatorname{sgn}(0) = -h^2 \lambda_1 L [-1, 1] && \Leftrightarrow \\ & b_k = -h^2 \lambda_1 L \xi \qquad \text{for some } \xi \in [-1, 1] && \Rightarrow \\ & \xi = \frac{b_k}{-h^2 \lambda_1 L}, \end{aligned} \tag{130}$$

  where ξ is called a selection of the set-valued signum function at zero.

- **Case 3**: $b_k > h^2 \lambda_1 L$

  Considering Fig. 2, $\operatorname{sgn}(\sigma_{0,k+1}) = -1$ is obtained for this case. Hence,

$$\sigma_{0,k+1} = h^2 \lambda_1 L - b_k. \tag{131}$$

The algorithm related to the second semi-implicit scheme is illustrated in Fig. 13. It can be seen that b_k depends on λ_0. As the result, the differentiator is sensitive to the differentiation gain λ_0 for the case 2. We remind that for the full implicit scheme the differentiator is insensitive to the gains for the case 2.

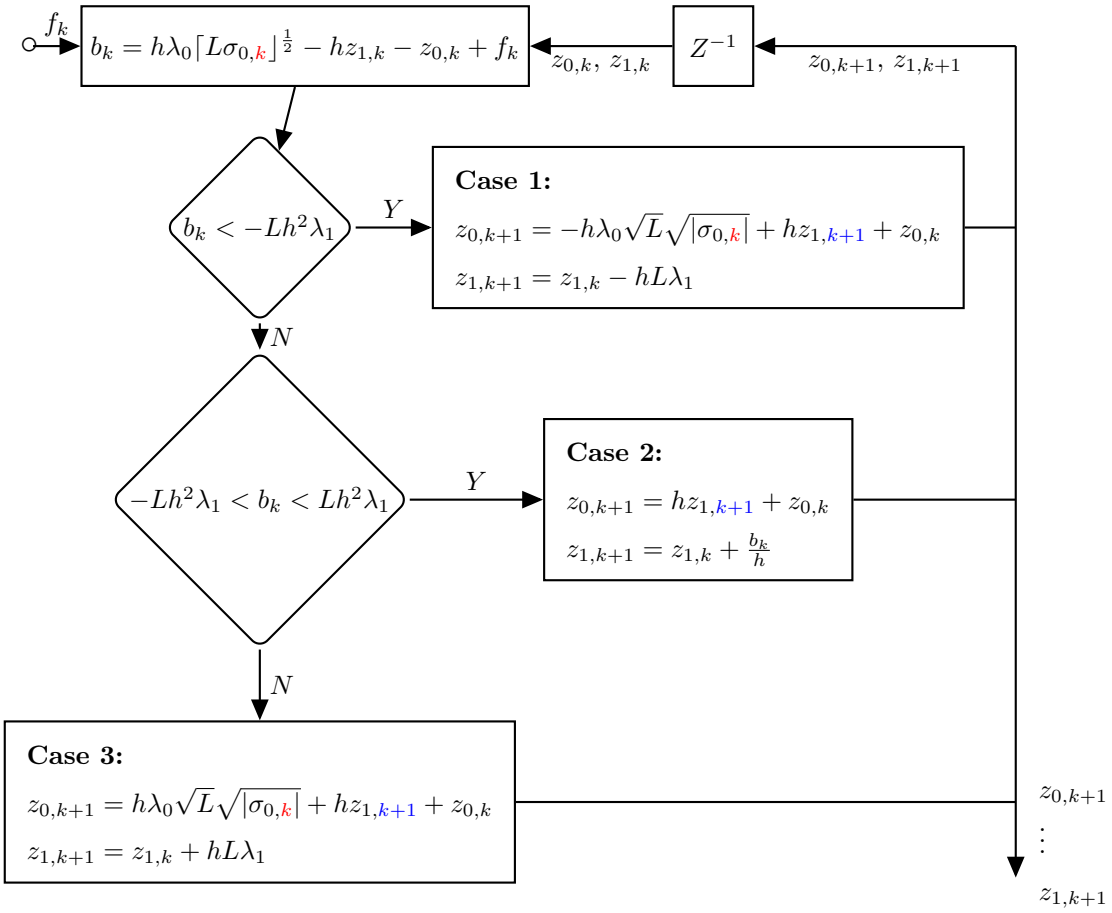

Figure 13: Flowchart of the second semi-implicit scheme for the STD in (125). The block Z^{-1} indicates one-step delay.

According to this semi-implicit scheme, the GE for the AO-STD can be written as follows:

$$\begin{cases} \sigma_{0,k+1} + b_k \in -h^{n+1}\lambda_n L\,\mathrm{sgn}(\sigma_{0,k+1}) \\ b_k = -\sum_{l=0}^{n} h^l z_{l,k} + f_k + \sum_{l=0}^{n-1}\left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}}\lceil\sigma_{0,k}\rfloor^{\frac{n-l}{n+1}}\right). \end{cases} \tag{132}$$

In this case, the fundamental operator of this semi-implicit scheme is given by (133).

$$x \mapsto \left(I_d + h^{n+1}\lambda_n L\,\mathrm{sgn}(\cdot)\right)^{-1}(-b_k) \tag{133}$$

It is clear that due to the fact that the terms $\lceil\sigma_{0,k}\rfloor^{\frac{n-1}{n+1}}$ are kept explicit, allows one to formulate the GE (132) which has the same form as the GE in (127). This semi-implicit AO-STD can be implemented as follows:

- **Case 1**: $b_k < -Lh^{n+1}\lambda_n$

  From Fig. 15, $\text{sgn}(\sigma_{0,k+1}) = 1$ is obtained. Thus,

$$\sigma_{0,k+1} = -h^{n+1}\lambda_n L - b_k. \tag{134}$$

- **Case 2**: $-Lh^{n+1}\lambda_n < b_k < Lh^{n+1}\lambda_n$

  In this case, one has $\sigma_{0,k+1} = 0$ as the only solution of the generalised equation in (132). Hence, the selection of the set-valued signum function is as follows:

$$\begin{aligned} & b_k \in -h^{n+1}\lambda_n L\text{sgn}(0) = -h^{n+1}\lambda_n L[-1,1] && \Leftrightarrow \\ & b_k = -h^{n+1}\lambda_n L\xi \qquad \text{for some } \xi \in [-1,1] && \Rightarrow \\ & \xi = \frac{b_k}{-h^{n+1}\lambda_n L}, \end{aligned} \tag{135}$$

  where ξ is called a selection of the set-valued signum function at zero. Note that in this case, (124) gives

$$\begin{aligned} & z_{i,k+1} = hz_{i+1,k+1} + z_{i,k} \\ & z_{n,k+1} = z_{n,k} + \tfrac{b_k}{h^n}, \qquad i = 0,\ldots,(n-1). \end{aligned} \tag{136}$$

- **Case 3**: $b_k > Lh^{n+1}\lambda_n$

  Considering Fig. 2, $\text{sgn}(\sigma_{0,k+1}) = -1$ is obtained for this case. Hence,

$$\sigma_{0,k+1} = h^{n+1}\lambda_n L - b_k. \tag{137}$$

The flowchart of the semi-implicit discretization for AO-STD is as follows:

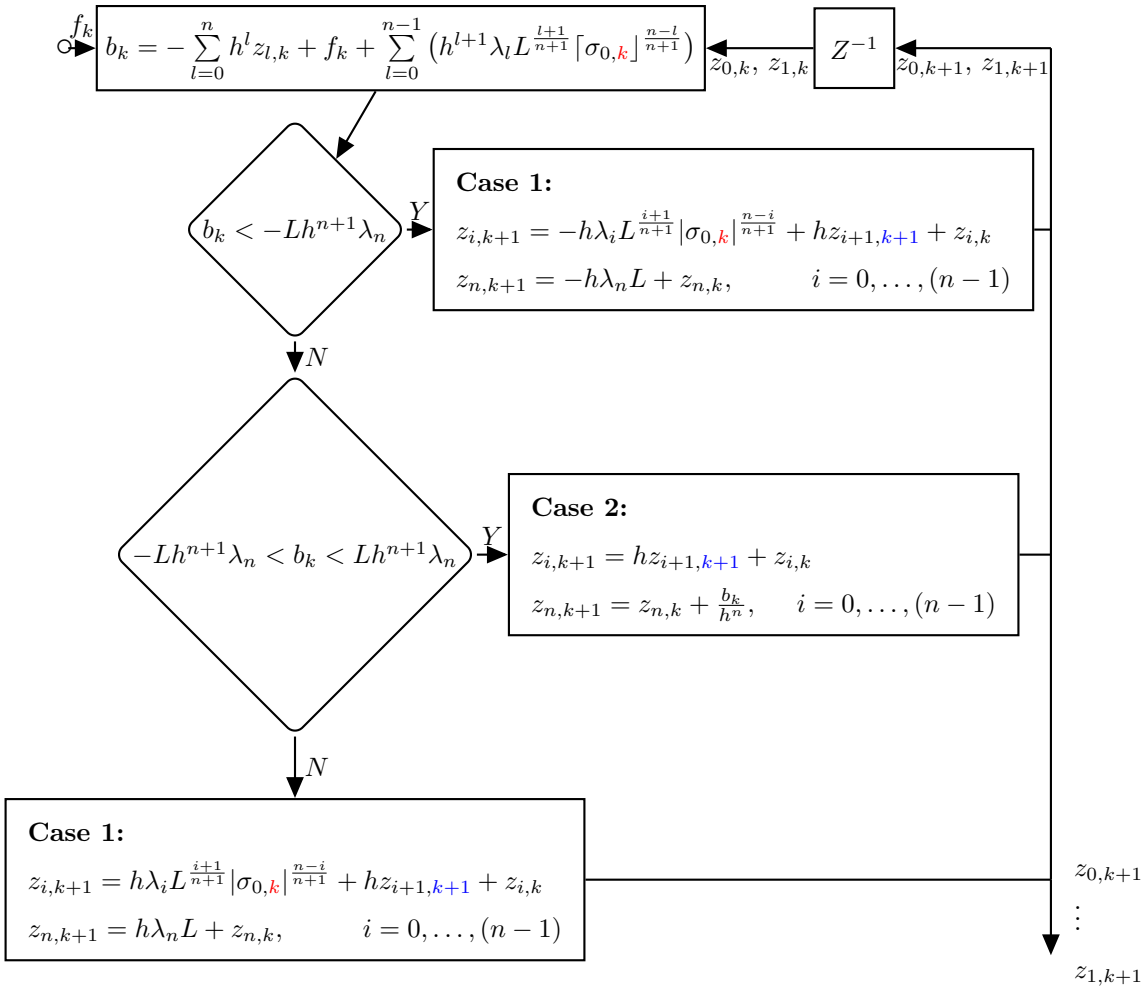

Figure 14: Flowchart of the second semi-implicit AO-STD (124). The block Z^{-1} indicates one-step delay.

According to Figs. 13 and 14, it can be seen that λ_n (for STD λ_1) does not appear in (136) for the case 2. However, other parameters still appear in b_k. Note that only the full implicit scheme is insensitive to the differentiation gains during the mode of case 2.

Here, the "fundamental" operators associated with each implicit/semi-implicit scheme are summarized in Table 1, where I_d stands for the identity function, i.e., $x \mapsto x$, and $(\cdot)^{-1}$ stands for the inverse of mapping, possibly set-valued.

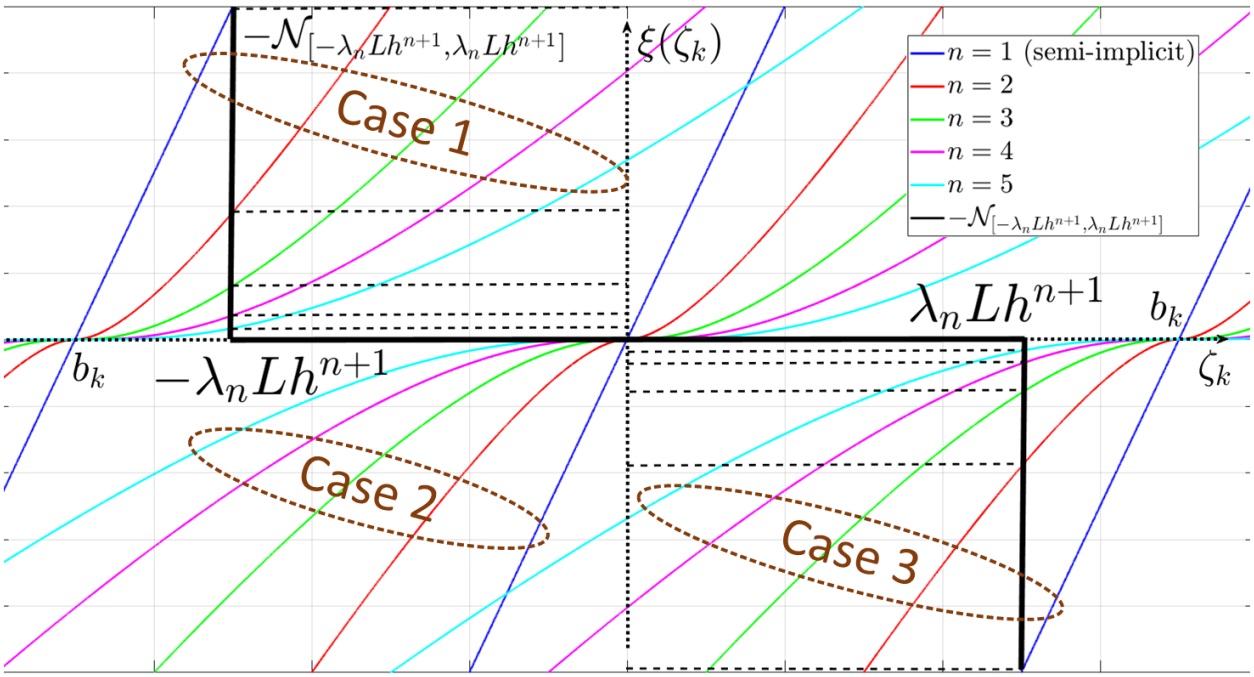

Figure 15: Graphical interpretation of I-STD ($n=2$) (45), I-URED ($n=4$) (63), I-AO-STD ($n=1,2,\dots$) (76), I-HDD (n=1,2,...) (98), I-GHDD (n=1,2,...) (108), I-FDFF and I-AO-FDFF ($n=1$) (112), SI-STD (n=1) (127), and SI-AO-STD (n=1) (132). Note that for the Case 2, the plots are only provided for $b_k=0$.

I-STD (45):	$\sigma_{0,k+1} \mapsto \left(I_d + a\lceil\cdot\rfloor^{\frac{1}{2}} + L\lambda_1 h^2 \operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
I-URED (63):	$\sigma_{0,k+1} \mapsto \left(I_d + h\lambda_0 L^{\frac{1}{2}}\left(\lceil\cdot\rfloor^{\frac{1}{2}} + \mu\lceil\cdot\rfloor^{\frac{3}{2}}\right) + h^2\lambda_1 L\left(\frac{1}{2}\operatorname{sgn}(\cdot) + 2\mu(\cdot) + \frac{3}{2}\mu^2\lceil\cdot\rfloor^2\right)\right)^{-1}(-b_k)$
I-AO-STD (76):	$\sigma_{0,k+1} \mapsto \left(I_d + \sum_{l=0}^{n-1}\left(h^{l+1}\lambda_l L^{\frac{l+1}{n+1}}\lceil\cdot\rfloor^{\frac{n-l}{n+1}}\right) + h^{n+1}\lambda_n L\operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
I-HDD (98):	$\sigma_{0,k+1} \mapsto \left(I_d + \sum_{l=0}^{n-1}\left(m_l h^{l+1}\lambda_l L^{\frac{l+1}{n+1}}\lceil\cdot\rfloor^{\frac{n-l}{n+1}}\right) + h^{n+1}\lambda_n L m_n \operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
I-GHDD (108):	$\sigma_{0,k+1} \mapsto \left(I_d - \sum_{i=0}^{n-1}\left(h^{i+1}\psi_{i,k+1}(\cdot)\right) + h^{n+1}L\lambda_n \operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
I-FDFF (112): I-AO-FDFF (112):	$\sigma_{0,k+1} \mapsto \left(aI_d + c\operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
SI-STD (127):	$\sigma_{0,k+1} \mapsto \left(I_d + h^2\lambda_1 L\operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$
SI-AO-STD (132):	$\sigma_{0,k+1} \mapsto \left(I_d + a\lceil\cdot\rfloor^{\frac{1}{2}} + L\lambda_1 h^2 \operatorname{sgn}(\cdot)\right)^{-1}(-b_k)$

Table 1: Fundamental operators associated with each implicit/semi-implicit scheme

Considering the fundamental operators related to all implicit and semi-implicit schemes, one can conclude that all the implicit and semi-implicit schemes follow the common structure $\sigma_{0,k+1} \mapsto \left(I_d + NL +\right.$

$\alpha\,\mathrm{sgn}(\cdot)\big)^{-1}(-b_k)$, where NL stands for nonlinear terms and $\alpha > 0$ is a constant. However, it should be mentioned that the terms b_k are not the same form one algorithm to the next.

Fig. 15 is an extension of Fig. 2 in a broader context, which depicts how the fundamental operators are solved by computing the intersection between the normal cone and the nonlinear terms, for various cases of AO-STD. The other operators are calculated similarly.

Definition 3 *The sliding-surface of the AO-STD is defined as follows:*

$$\Sigma_d \triangleq \{(z_{0,k},\dots,z_{n,k}) \in \mathbb{R}^n \mid z_{0,k} = f_k^{(0)}, z_{1,k} = f_k^{(1)}, z_{2,k} = f_k^{(2)}, \dots, z_{n-1,k} = f_k^{(n-1)}, z_{n,k} = f^{(n)}\}. \quad (138)$$

Lemma 5 *Assuming $f^{(n+1)}(t) = 0$, the sliding-surface as in Definition 3 is invariant for the I-AO-STD (Fig. 7), as well as the SI-AO-STDs (Fig. 14 and (122)) during the sliding-phase (Case 2).*

Proof Considering $f_k^{(n+1)} = 0$, the AO-STD takes the homogeneous form (see Remark 6). Substituting $z_{i,k}, i = 0,\dots,n$ from (138) into second case of Figs. 7 and 14 and (122) yields, $z_{i,k+1} = z_{i,k}, i = 0,\dots,n$. Hence, I-AO-STD as well as the semi-implicit counterparts are invariant during the sliding-phase.

3.4.3 Semi-implicit differentiators in the literature

As far as the authors have investigated, the only semi-implicit differentiator has been recently introduced in [69], where a semi-implicit discretization method is proposed for the URED (8) as follows:

$$\begin{cases} z_{0,k+1} = \big(-1 + \hat{\psi}_{0,k}(\sigma_{0,k})\big)\sigma_{0,k} + hz_{1,k} + f_k & (139a) \\ z_{1,k+1} = \dfrac{-1 + \hat{\psi}_{1,k}(\sigma_{0,k})}{h} + \sigma_{0,k} + z_{1,k}, & (139b) \end{cases}$$

where

$$\begin{cases} \hat{\psi}_{0,k} = \dfrac{hL^{\frac{1}{2}}\lambda_0\mu|\sigma_{0,k}|^{\frac{3}{2}} + 2|\sigma_{0,k}| + hL^{\frac{1}{2}}\lambda_0|\sigma_{0,k}|^{\frac{1}{2}}}{\frac{3}{2}h^2L\lambda_1\mu^2|\sigma_{0,k}|^2 + hL^{\frac{1}{2}}\lambda_0\mu|\sigma_{0,k}|^{\frac{3}{2}} + hL^{\frac{1}{2}}\lambda_0|\sigma_{0,k}|^{\frac{1}{2}} + (1+2h^2L\lambda_1\mu)|\sigma_{0,k}| + \frac{1}{2}h^2L\lambda_1} & (140a) \\ \hat{\psi}_{1,k} = \dfrac{hL^{\frac{1}{2}}\lambda_0\mu|\sigma_{0,k}|^{\frac{3}{2}} + |\sigma_{0,k}| + hL^{\frac{1}{2}}\lambda_0|\sigma_{0,k}|^{\frac{1}{2}}}{\frac{3}{2}h^2L\lambda_1\mu^2|\sigma_{0,k}|^2 + hL^{\frac{1}{2}}\lambda_0\mu|\sigma_{0,k}|^{\frac{3}{2}} + hL^{\frac{1}{2}}\lambda_0|\sigma_{0,k}|^{\frac{1}{2}} + (1+2h^2L\lambda_1\mu)|\sigma_{0,k}| + \frac{1}{2}h^2L\lambda_1}. & (140b) \end{cases}$$

It is shown in [43] that (139) is globally asymptotically stable for the following condition:

$$\begin{cases} \lambda_0 > \dfrac{0.397945}{h} & (141a) \\ \lambda_1 \geq \dfrac{\lambda_1^2}{2} & (141b) \\ L \geq 1 & (141c) \\ \mu \geq 1. & (141d) \end{cases}$$

Moreover, $\frac{1}{h}$ should be at least ten-times larger than the largest frequency component of the input signal f_k in Hertz. To tune the parameters of (140), a unified parameter tuning algorithm will be introduced in Section 4.

3.5 Kalman differentiator

The literature shows, unlike the above mentioned differentiators, there are also a few number of differentiators which are designed directly in the discrete-time manner. One of th well-known of such a differentiator is designed based on the discrete-time Kalman algorithm. In this case, the differentiation problem should be formulated in the discrete-time state-space equations of a chain of integrators as follows [31]:

$$\begin{cases} z_{k+1} = Az_k & \text{(142a)} \\ y_k = f_k - Cz_k, & \text{(142b)} \end{cases}$$

where $z_k = [z_{0,k}, z_{1,k}, \ldots, z_{n,k}]^T \in \mathbb{R}^{n+1}$ is the estimation vector, $z_{i,k}$ is the estimation of the i-th order differentiation of the input f_k at the time step k, and y_k is called the Kalman innovation. The parameters for a third-order Kalman differentiation is as follows:

$$A = \begin{bmatrix} 1 & h & \frac{h^2}{2} & \frac{h^3}{3!} \\ 0 & 1 & h & \frac{h^2}{2} \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}. \tag{143}$$

Following the standard Kalman algorithm, the covariance matrices of the process and the measurement noises are shown by Q and R, respectively, where

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h \end{bmatrix}, \qquad R = \sigma^2. \tag{144}$$

Here, the scalar parameter σ can be obtained according to the optimization algorithm presented in Section 4. Afterwards, the differentiation can be performed according to the standard Kalman algorithm as follows (see [70] for the algorithm as well as the preliminary equations of the Kalman filter):

- **Predict:**
  1. Predicted state estimate: $z^*_{k+1} = Az_k$
  2. Predicted estimate: covariance: $P^*_{k+1} = AP_kA^T + Q$
- **Update:**
  3. Innovation residual: $y_k = f_k - Cz_k$
  4. Innovation covariance: $S_k = CP^*_{k+1}C^T + R$
  5. Optimal Kalman gain: $K_k = P^*_{k+1}C^TS_k^{-1}$
  6. Updated state estimate: $z_{k+1} = z_k + K_ky_k$
  7. Updated estimate covariance: $P_{k+1} = P^*_{k+1} - K_kCP^*_{k+1}$

where $P_0 = \mathbf{0}$.

4 Tuning the parameters of the differentiators

To provide a fair comparison among all the differentiators which are introduced in the foregoing sections, a systematic tuning procedure is presented in this section.

4.1 Objective functions

Some objective functions are defined to evaluate the differentiators. In the following equations, e_k denotes the differentiation error, and y_k is the output of a differentiator. In the following notation, t_f denotes the simulation time, and N is the number of discrete-time steps in $[0, t_f]$, with $h = \frac{t_f}{N}$. The discrete-time steps can also be defined as $t_k = hk$ for $k = 0, 1, \ldots, N$.

- **Average magnitude of differentiation error**: This cost function indicates the average energy of the differentiation error. Smaller amount of this value indicates better exactness. This cost function is defined as follows:

$$\bar{L}_2(e_k) = \frac{h}{t_f}||e_k|| = \frac{h}{t_f}\sqrt{\sum_{k=0}^{t_f/h} e_k^2}, \tag{145}$$

  where e_k denotes the differentiator error.

- **Magnitude of the interpolated differentiation error**: In some cases, it is necessary to compare the results for different sampling periods. In this case, it would be more logical to interpolate the discontinuous error signal using an interpolation scheme, before calculating the energy of the differentiation error as follows:

$$\tilde{L}_2(e_k) = ||\tilde{e}_k|| = \sqrt{\sum_{k=0}^{t_f/h_c} \tilde{e}_k^2}, \tag{146}$$

  where $\tilde{e}_k$ denotes the interpolated differentiator error, and h_c is the sampling period of the interpolated error ($h_c < h$). Here, $\tilde{e}_k$ is calculated using linear interpolation. MATLAB command `vq = interp1(x,v,xq,method)` can be used to calculate $\tilde{e}_k$ where `x` is the time vector of e_k, `v` is the vector of differentiation error, `xq` is the new time vector for the interpolated differentiation error, `vq` is the interpolated differentiation error, and `method` indicates the interpolation method. In this study, linear interpolation is used for this purpose, i.e., `method='linear'`. Note that the lengthes of `x` and `v` are t_f/h, while the lengthes of the interpolated vector `vq` and the new time vector `xq` are t_f/h_c, where t_f is the simulation time. Since $h_c < h$, the length of the interpolated error `vq` is higher than that of `v`. Throughout the manuscript, $h_c = 100\mu$s is considered to calculate $\tilde{L}_2(e_k)$.

- L_∞ **norm of the differentiation error**: This cost function indicates the maximum deviation of the differentiator output from the real differentiation. Thus, it can be used to calculate the overshoots and the accuracy of the differentiation algorithm. This cost can be calculated as follows:

$$L_\infty(e_k) = ||e_k||_\infty = \max_k |e_k|, \quad k = 0, \ldots, t_f/h. \tag{147}$$

- **Total variation of error**: This criterion can be used to qualify the chattering effect. It is calculated as follows:

$$\mathrm{VAR}(y_k) = \sum_{k=0}^{t_f/h} |y_k - y_{k-1}|. \tag{148}$$

- **Total harmonic distortion (THD)**: Unlike the previous performance functions, which are defined for the time-domain, this one is defined to analyze the frequency response of a differentiator. THD illustrates how a discretization method can preserve the frequency specifications of a sinusoidal signal. THD is defined as follows:

$$\mathrm{THD}(y_k) = 100\frac{\sqrt{\sum_k V_k^2}}{V_0}, \quad k = 0, \ldots, t_f/h, \tag{149}$$

  In this study, it is assumed that the input signal of the differentiators is a pure sinusoidal signal with one frequency component V_0. Hence, considering V_k as the frequency components of the output of a differentiator, (149) indicates the harmonic's level generated by a differentiator. It should be noted that harmonics are considered as undesired phenomena and are caused by an inappropriate discretization scheme. Therefore, a smaller amount of THD illustrates a better response.

4.2 Problem formulation

Problem formulation for the optimization of the differentiators' parameters can be defined as follows. Note that J denotes the index function, and in this work we have set $J = 10000\bar{L}_2(e_k)$. The reason for this selection is that the $\bar{L}_2$ norm of the error indicates the ability of a differentiator to track an input. In other words, the tuning procedure is conducted based only on the cost $\bar{L}_2$. Other objective functions could be used, for the purpose of monitoring, to evaluate the performance of the differentiators in the next sections.

- **STD and AO-STD**:

$$\begin{aligned} \underset{L}{\text{minimize}} \quad & J(L) \\ \text{subject to} \quad & L > 0 \end{aligned} \tag{150}$$

- **URED**:

$$\begin{aligned} \underset{\mu,L}{\text{minimize}} \quad & J(\mu, L) \\ \text{subject to} \quad & \mu, L > 0 \end{aligned} \tag{151}$$

- **QD**:

$$\begin{aligned} \underset{F,\alpha}{\text{minimize}} \quad & J(F,\alpha) \\ \text{subject to} \quad & F,\alpha > 0 \end{aligned} \tag{152}$$

- **LF**:

$$\begin{aligned} \underset{c}{\text{minimize}} \quad & J(c) \\ \text{subject to} \quad & c > 0 \end{aligned} \tag{153}$$

- **ALIEN**:

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & J(T) \\ \text{subject to} \quad & T > 0;\ \kappa,\mu \in \mathbb{N} \end{aligned} \tag{154}$$

- **HD**:

$$\begin{aligned} \underset{r}{\text{minimize}} \quad & J(r) \\ \text{subject to} \quad & r > 0 \end{aligned} \tag{155}$$

- **HDD**:

$$\begin{aligned} \underset{L}{\text{minimize}} \quad & J(L) \\ \text{subject to} \quad & L > 0 \end{aligned} \tag{156}$$

- **GHHD**:

$$\begin{aligned} \underset{L}{\text{minimize}} \quad & J(L) \\ \text{subject to} \quad & L > 0 \end{aligned} \tag{157}$$

- **VGED**:

$$\begin{aligned} \underset{\mu,\tau,q,\omega_c}{\text{minimize}} \quad & J(\mu,\tau,q,\omega_c) \\ \text{subject to} \quad & \mu,\tau,q,\omega_c > 0 \end{aligned} \tag{158}$$

- **STDAC**:

$$\begin{aligned} \underset{\alpha,\epsilon}{\text{minimize}} \quad & J(\alpha,\epsilon) \\ \text{subject to} \quad & 0 < \alpha < \lambda_0, \epsilon > 0 \end{aligned} \tag{159}$$

- **FDFF**:

$$\begin{aligned} \underset{\omega_s,\omega_f,\rho,\gamma}{\text{minimize}} \quad & J(\omega_s,\omega_f,\rho,\gamma) \\ \text{subject to} \quad & \omega_s,\omega_f,\gamma > 0,\ \rho > 1 \end{aligned} \tag{160}$$

- **AO-FDFF**:

$$\begin{aligned} \underset{F,\epsilon,\omega_s,\omega_f,\alpha_1,\rho}{\text{minimize}} \quad & J(F,\epsilon,\omega_s,\omega_f,\alpha_1,\rho) \\ \text{subject to} \quad & F,\epsilon,\omega_s,\omega_f,\alpha_1 > 0,\ \rho > 1 \end{aligned} \tag{161}$$

- **Kalman**:

$$\begin{aligned} \underset{R}{\text{minimize}} \quad & J(R) \\ \text{subject to} \quad & R > 0 \end{aligned} \tag{162}$$

Since the optimization problem depends on the noise and nonlinear dynamics, it can be considered as a nonlinear stochastic problem. In this research, an interval has been found for each parameter based on the constraints. Afterward, a randomized optimization algorithm [71] has been utilized to calculate the optimal parameters. Finally, the best combination of parameters has been selected as the solution. Other optimization methods may also be utilized, but it goes out of the scope of this work.

Note that the optimal values for the differentiation gains λ_i in (7), (8) and (12) are obtained using numerical simulations and were given in [10]. These parameters are presented in Table 2 (see Remarks 14 and 15 for comments about gain tuning).

Table 2: Constant parameters of the exact differentiators

Order	λ_0	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
0	1.1							
1	1.5	1.1						
2	2	2.12	1.1					
3	3	4.16	3.06	1.1				
4	5	10.03	9.30	4.57	1.1			
5	7	23.72	32.24	20.26	6.75	1.1		
6	10	47.69	110.08	101.96	43.65	9.91	1.1	
7	12	84.14	281.37	455.40	295.74	88.78	14.13	1.1

Remark 9 SNR *is the signal-to-noise ratio and is defined in (163). In order to add a white noise to the input signal, in this study, MATLAB command* `y=awgn(x,SNR,'measured')` *has been utilized where* `x` *is the input signal and* `y` *is the polluted signal.*

One has:

$$\mathrm{SNR} = 10\log\left(\frac{Q_S}{Q_n}\right), \tag{163}$$

where SNR is in decibel (dB), Q_s and Q_n are powers of the signal and noise, respectively. Here, the power is equal to the variance of the signal.

Remark 10 *In this study, the L_2 norm of the differentiation error is considered as the objective function for the optimization. Several types of objective functions like L_∞ of the estimation error can also be considered. One may redo all the simulations based on different objective functions using the toolbox which is developed in this work (see Appendix E).*

Remark 11 *The optimization method which is utilized in this work may not be efficient enough to find the optimal parameters. Hence, a sensitive differentiator may not be able to show appropriate responses. It leads to identifying a sensitive differentiator.*

5 Open-loop analysis of the differentiators

The performance of the differentiators in open-loop configuration will be considered in this section using numerical simulations in MATLAB environment. In order to provide a fair comparison, the tuning procedure (which is presented in Section 4) has been conducted to calculate sub-optimal[1] parameters which are shown in Table 3. Investigations show that the parameters should be designed for a noisy condition. Otherwise, the optimization algorithm will select large differentiation gains, which deteriorate the robustness of the

[1]The tuning procedure is based on a heuristics algorithm whose optimality is not guaranteed.

differentiators with respect to noise. Thus, a relatively small SNR=30dB is considered for this optimization problem.

Table 3: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2(e_k)$
Euler (3)	No parameter	400.7426
LF (4)	c=7.1113	114.5675
E-STD (21)	L=0.7713	92.7441
I-STD (Fig. 3)	L=0.7324	87.1980
SI-STD (123)	L=0.6985	101.2067
E-URED (26)	L=0.0770, μ=20.8386	86.7613
I-URED (Fig. 5)	L=0.1021, μ=21.2075	82.9217
E-QD (25)	F=4.4026, α= 0.3780	102.6753
I-QD (60)	F=4.5323, α=0.8123	104.6224
ALIEN (14)	T=0.7025 , κ=1 , μ=4	46.9701
HD (40)*	r=1.7034	81.6405
E-AO-STD (22)**	L=4.8973	93.1914
I-AO-STD (Fig. 7) **	L=2.9122	47.9806
SI-AO-STD (122)**	L=2.8157	75.5441
E-HDD (24)**	L=4.9392	79.3572
E-GHDD (27)**	L=4.8970	77.8480
I-HDD (Fig. 9)**	L=2.9921	44.3107
I-GHDD (Fig. 10)**	L=2.9822	43.4911
VGED (35)	μ=4.3694, τ=1.3269, ω_c=12.2205, q=0.2997	89.2798
SI-URED (139)	L=0.1434, μ=93.5748	94.3047
E-STDAC (10)	α=0.5318, ϵ=0.0000	89.5387
I-FDFF (Fig. 12)	ω_s=19.6607, ω_f=8.4727, ρ=8.6929, γ=0.0348	95.9795
I-AO-FDFF (Fig. 12)	F=37.7845, ϵ=18.6061, ω_s=2.5068	50.2447
	ω_f=62.6396, α_1=456.7015, ρ=88.3003	
Kalman (Section 3.5)	$R = 8.4121 \times 10^{-4}$	51.9665
HGD (16)	L=3.4554	95.7331
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]).		
** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation		
Input signal: $\sin(t)$, Output: first-order differentiation		
Noise type: white, SNR=30dB, h = 50ms. Performance: red<black<blue		

From Table 3, it can be seen that the higher-order implicit schemes, i.e., I-AO-STD, I-HDD, and I-GHDD, as well as I-AO-FDFF, as well as the ALIEN have achieved the smallest costs. Therefore, as a first impression, these differentiators can present the smallest $J = 10000\bar{L}_2$, at least for these specific conditions (SNR=30dB, h=50ms). On the other hand, the Euler method shows the largest J because it is too sensitive to the input noise.

Table 3 shows that L=0.7324 is obtained for the I-STD. Since $|\sin((k+1)h) - \sin(kh)| < 0.05$, $h = 0.05$s, and $\lambda_1 = 1.1$, its parameter should be selected such that $L > \frac{|\sin((k+1)h) - \sin(kh)|}{h^2\lambda_1} = 18.2$ to keep the I-STD on its sliding-surface (87) and therefore, ensure the exactness for the input signal $f_0(t) = \sin(t)$ (see (91a)). However, a smaller L is selected by the optimization procedure to also satisfy (91b) and therefore cancel the exactness for the noise. Thus, in this case, the optimization procedure is in accordance with Remark 7.

Considering Table 3, it can be seen that the gain of the HGD is even less than that of the E-AO-STD. In fact, because of the noise, the tuning algorithm selected a relatively small gain for the HGD to reduce its sensitivity to the noise.

5.1 Performances in noise-free conditions

A noise-free condition is selected to investigate the numerical chattering of the differentiators with the parameters shown in Table 3. The output of the differentiators and the corresponding error are shown in Figs. 16 and 17, respectively. Furthermore, the results are summarized in Table 4. Let us remind that all the references for all differentiators definitions are reported in Table 3.

Remark 12 *The calculation time indicates the required time to calculate the differentiation for the overall simulation time. Assuming that the simulation time is 10s, and $h = 50ms$, there will be 200 time-steps. The calculation time for the overall 200 steps is shown in Table 4. The simulations are conducted on* `Intel Core i3-4030` *CPU with two cores working at 1.9 GHz. MATLAB commands* `tic` *and* `toc` *are used to calculate this time. So, the calculation time might not be exact.*

The following results have been achieved for the open-loop tests:

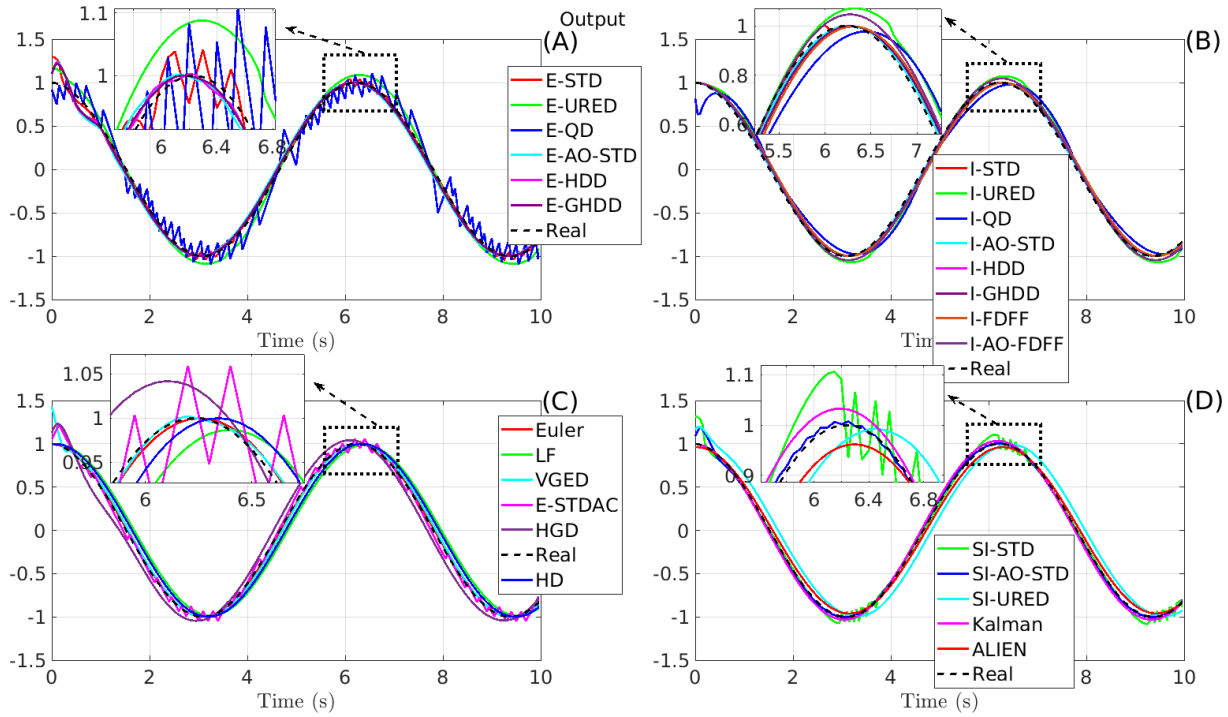

Figure 16: First-order differentiation under a noise-free condition. Parameters are shown in Table 3 ($h =$ 50ms).

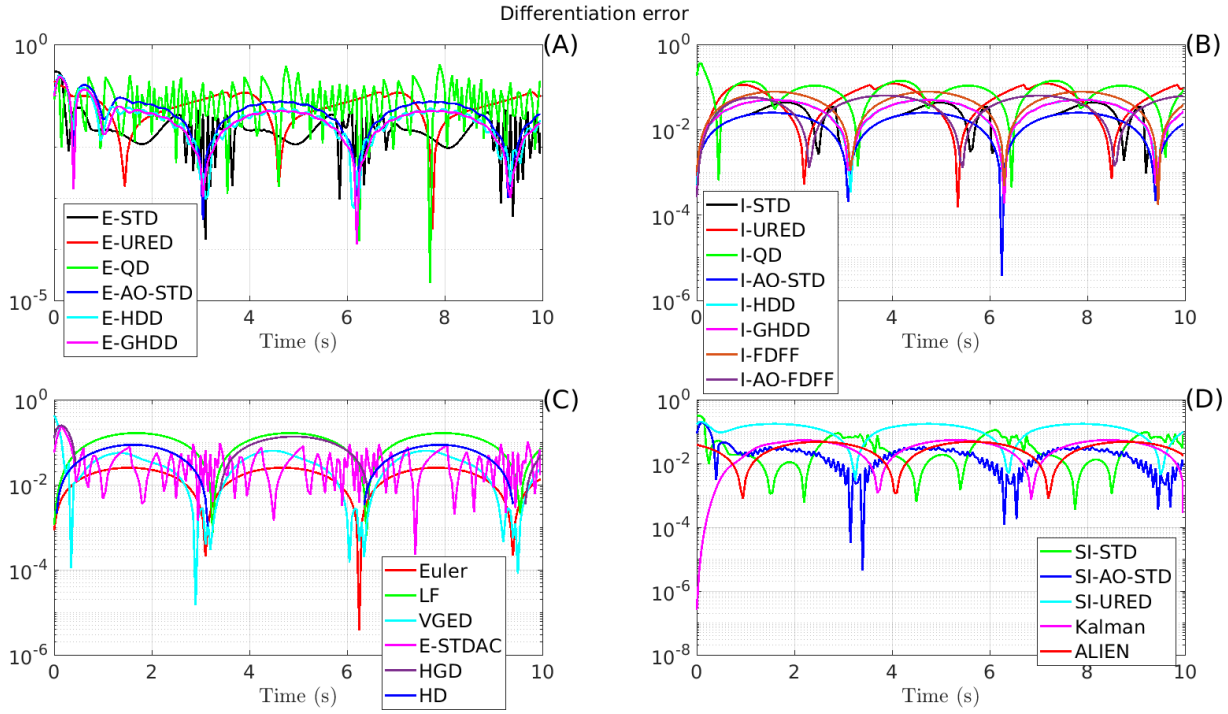

Figure 17: Error of the first-order differentiation under a noise-free condition. Parameters are shown in Table 3 ($h = 50$ms).

Table 4: Results for the first-order differentiation under a noise-free condition

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0012	0.0273	0.0250	6.1735	4.4747	1.00 β
LF	0.0080	0.1778	0.1634	6.0366	4.4961	1.05 β
E-STD	0.0034	0.0707	0.2958	7.9831	4.8680	1.25 β
I-STD	0.0018	0.0393	0.0436	6.2672	4.4556	1.87 β
SI-STD	0.0043	0.0847	0.3199	10.4075	4.9141	1.27 β
E-URED	0.0056	0.1228	0.1912	6.8031	4.5902	1.68 β
I-URED	0.0050	0.1115	0.1209	6.5166	4.3794	52.12 β
E-QD	0.0088	0.1776	0.4070	24.0345	4.8868	1.62 β
I-QD	0.0069	0.1536	0.3632	6.2721	4.2441	2.78 β
ALIEN	0.0024	0.0527	0.0476	5.9185	3.3207	21.10 β
HD	0.0041	0.0916	0.0868	6.1300	3.4203	5.80 β
E-AO-STD	0.0048	0.1075	0.2519	6.6146	4.7046	3.34 β
I-AO-STD	0.0012	0.0273	0.0250	6.1731	4.4744	7.20 β
SI-AO-STD	0.0024	0.0535	0.1993	6.5635	4.6337	3.47 β
E-HDD	0.0039	0.0853	0.2496	6.6105	4.6926	3.60 β
E-GHDD	0.0037	0.0829	0.2338	6.5664	4.6844	4.49 β
I-HDD	0.0024	0.0544	0.0505	6.1533	4.4770	35.18 β
I-GHDD	0.0024	0.0544	0.0506	6.1537	4.4767	34.45 β
VGED	0.0038	0.0781	0.4293	6.6327	4.9070	23.72 β
SI-URED	0.0091	0.2033	0.2036	6.2698	4.7269	1.67 β
E-STDAC	0.0036	0.0752	0.2404	11.0826	4.6536	1.78 β
I-FDFF	0.0038	0.0837	0.0779	6.1329	4.4791	2.20 β
I-AO-FDFF	0.0030	0.0675	0.0631	6.4028	4.4212	21.07 β
Kalman	0.0026	0.0580	0.0539	6.3832	4.4295	4.78 β
HGD	0.0075	0.1680	0.2476	6.8306	4.6807	2.45 β
h=50ms, and parameters are shown in Table 3.						
$\beta = 0.2275$ms. Performance: red<black<blue						

- **Explicit methods**: It is well-known that explicit methods suffer from the digital chattering problem in the context of sliding-mode control [53, 55, 56, 58, 72]. It is expected that a similar behavior occurs for differentiators. From Table 3, it can be seen that the optimization algorithm has selected small differentiation gains for the explicit differentiators to decrease the output chattering (output chattering increases $\bar{L}_2$). However, even with these small parameters, the explicit differentiators have

presented significant chattering, as seen in Fig. 16(A) and Fig. 17(A). According to Remark 3, unlike implicit methods, the explicit ones are sensitive to the differentiation gains during the sliding-phase. Thus, even in this noise-free case, the explicit methods presented a significant amount of numerical chattering. *Hence, the first recommendation is to avoid any explicit (forward) discretization of the exact differentiators.* Also, SI-STD in Fig. 16 (D) and Fig. 17 (D) is not good (worse than E-STD in Fig. 16 (A) and Fig. 17 (A)) while SI-AO-STD's performance (Fig. 16 (C) and Fig. 17 (C)) is situated in between that of E-AO-STD and I-AO-STD.

- **Linear and Euler filters**: It is well-known that in a noise-free condition, Euler method presents the best responses in the sense of the exactness. Since SNR=∞ is considered for this simulation, the effect of noise is completely neglected. Therefore, the Euler method has presented one of the best results. Since the only parameter of the LF is designed for a noisy condition (SNR=30dB), the optimization algorithm has selected a relatively small pole ($c = 7.1113$ in Table 3) to decrease the bandwidth. It can be seen that with this small bandwidth, the LF has presented a significant amount of phase-lag (see Fig. 16(C)). Another observation from Table 4 is that the HGD' results are comparable to those of the LF. Since the gain of the HGD is tuned for a noisy condition, the a small gain is calculated for the HGD by the tuning algorithm. Hence, the HGD behaves as a linear filter as expected.

- **ALIEN differentiator**: The parameters of this differentiator are tuned using the optimization procedure and shown in Table 3. The steady-state behavior of the ALIEN differentiator is quite good and comparable with that of the implicit sliding-mode algorithms in Figs. 16 and 17 (B) (see other simulations in Section 5.5) for transient responses.

- **Implicit methods**: Fig. 16 (B), Fig. 17 (B) and Table 4 show that the implicit methods (I-STD, I-AO-STD, I-HDD, I-GHDD, I-FDFF, and I-AO-FDFF) converge to the real differentiation without chattering (without or with much smaller phase-lag which appears in Fig. 16 for LF). It is interesting to note that the I-AO-STD has presented almost the same performance as the Euler method in this noise-free condition. Thus, I-AO-STD has presented almost the ideal response in the noise-free condition. Moreover, there is not any significant difference between the implicit and other methods in the case of the calculation time, except for the I-AO-STD, I-URED, I-HDD, I-GHDD and I-AO-FDFF which both require longer calculation time. As mentioned in Sections 3.3.3 and 3.3.4, in these differentiators, a polynomial equation must be solved at each time-step. Hence, the calculation time for the implicit discretization of these differentiators is around 10 times higher than its explicit counterpart. A more efficient numerical method may solve this issue. It is inferred from our research that the implicit methods supersede the explicit ones. *Therefore, implicit discretization can be considered as an appropriate solution for the differentiation task.*

5.2 Analysis of robustness against the noise

To investigate the robustness of the differentiators, their responses have been analyzed in the presence of white, sinusoidal, and bell-shaped types of noise in Section 5.2.1, Section 5.2.2, and Section 5.2.3, respectively.

5.2.1 Robustness against white noise

In this section, the input signal is polluted by white noise with SNR=30dB. The parameters are provided in Table 3, and the corresponding responses are shown in Figs. 18 and 19 and Table 5. It can be seen that noise has deteriorated all the results, as expected. From Fig. 18(C) and Fig. 19(C), it can be seen that the Euler method is not the best solution anymore, and even presents the worst response. According to the results, the implicit methods still present appropriate responses even in this noisy condition. Table 5 shows that I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF have presented the best responses. *Thus, the second recommendation of this work is to utilize I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF to achieve better robustness to noise.* In fact, here, the only drawback of the implicit methods is the higher calculation burden. Considering the results in Table 5 and Figs. 20 and 21 it can be seen that the semi-implicit methods (SI-AO-STD and SI-URED) also presented relatively good responses. Since the SI-URED and SI-AO-STD present a smaller calculation time than the I-AO-STD, it may be considered as the preferred option in the applications where the computational resources are limited. However, these semi-implicit schemes cannot provide the same performance as the I-AO-STD.

Comparing Fig. 18 (B) and (C) and based on Table 5, it can be concluded that I-STD, SI-AO-STD, I-URED, I-AO-STD, I-HDD, I-GHDD, I-FDFF, and I-AO-FDFF presented better responses than Euler and LF. Fig. 18(A), 19(A) and Table 5 show that among explicit methods, E-QD presented the worst responses. It also should be mentioned that the ALIEN method presented a good steady-state performance (see Section 5.5 for a further analysis of the ALIEN differentiator in transients).

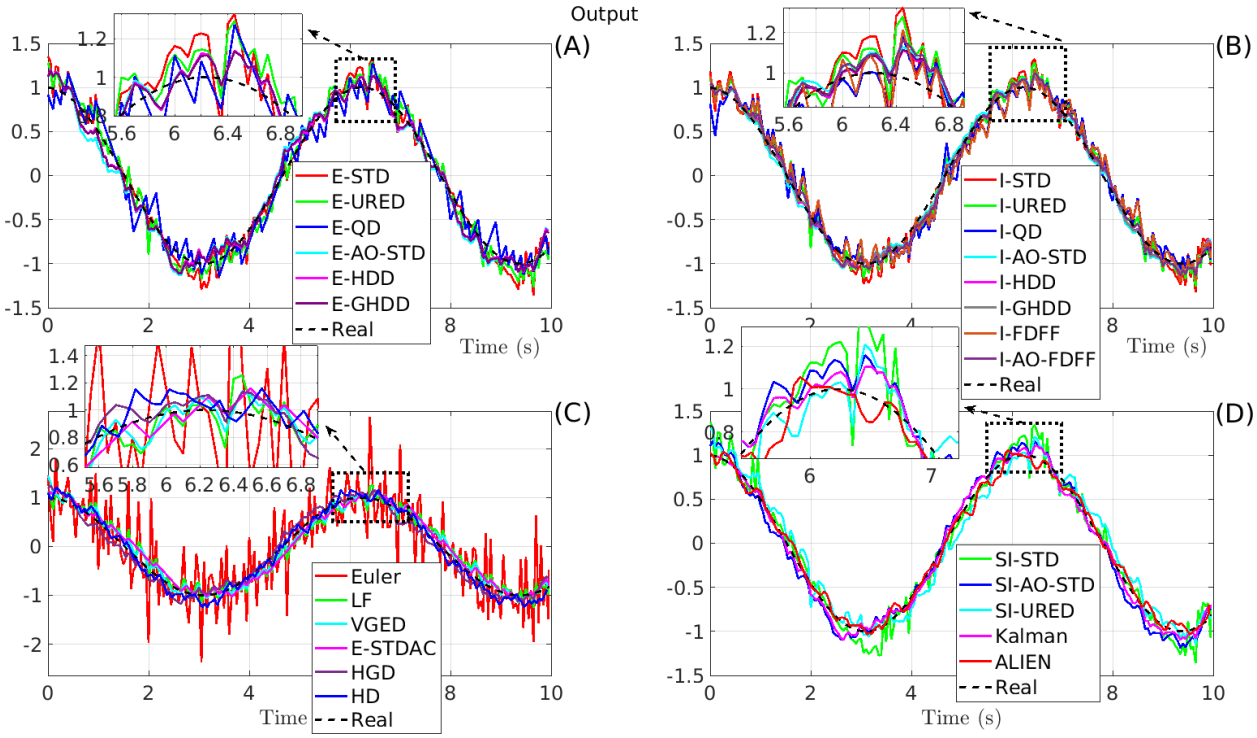

Figure 18: First-order differentiation in the presence of white noise. Parameters are shown in Table 3 ($h = 50$ms, SNR=30dB)

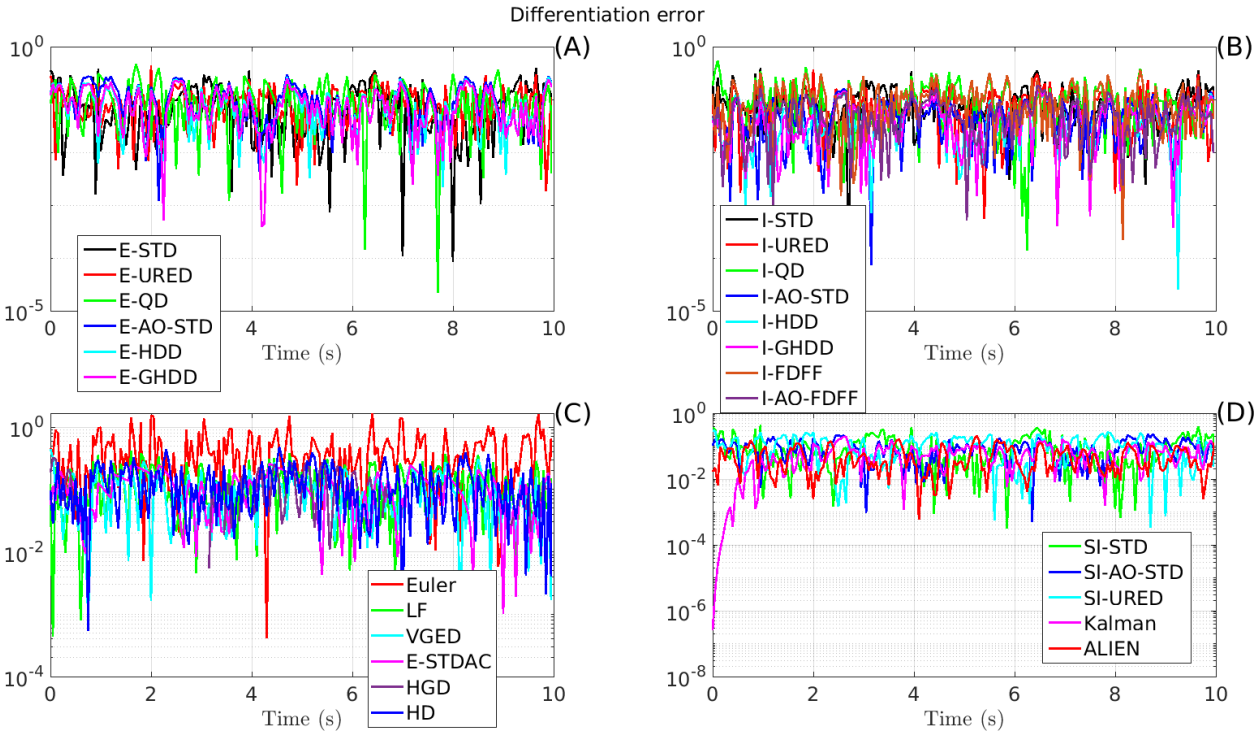

Figure 19: Error of the first-order differentiation in the presence of white noise. Parameters are shown in Table 3 ($h = 50$ms, SNR=30dB).

- FDFF and AO-FDFF: it is certainly in this case that superiority of the implicit discretization is the more sliding, compare Figs. 16 and 17 (B). The implicit method allows to drastically improve the Sliding-mode based differentiators performance, in particular the digital chattering (see VAR for I-FDFF in Table 4).

Table 5: Results for the first-order differentiation in the presence of white noise

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0401	0.6328	1.6678	156.1079	10.8251	1.00 β
LF	0.0115	0.2312	0.4237	27.2973	5.1186	1.39 β
E-STD	0.0093	0.1813	0.3950	22.6491	5.0380	1.75 β
I-STD	0.0087	0.1694	0.3991	22.3047	4.8264	1.79 β
SI-STD	0.0101	0.1971	0.4364	22.6797	5.0543	1.60 β
E-URED	0.0087	0.1733	0.4380	21.2378	4.9033	1.59 β
I-URED	0.0083	0.1647	0.3669	21.3538	4.7479	28.40 β
E-QD	0.0103	0.2137	0.4616	23.8976	4.8946	1.92 β
I-QD	0.0105	0.2165	0.5410	23.6339	4.7218	2.00 β
ALIEN	0.0047	0.1006	0.2083	8.3567	3.4701	27.62 β
HD	0.0082	0.1740	0.2664	15.5745	4.3390	8.03 β
E-AO-STD	0.0093	0.2025	0.2947	11.7065	4.7248	2.60 β
I-AO-STD	0.0048	0.1032	0.1565	8.6579	4.4372	27.27 β
SI-AO-STD	0.0076	0.1651	0.2333	10.3001	4.6059	3.65 β
E-HDD	0.0079	0.1707	0.2623	11.7419	4.6817	3.45 β
E-GHDD	0.0078	0.1682	0.2496	11.0980	4.6572	4.44 β
I-HDD	0.0044	0.0948	0.1454	8.7591	4.4111	27.19 β
I-GHDD	0.0043	0.0935	0.1420	8.4464	4.4002	24.47 β
VGED	0.0089	0.1889	0.4458	16.1570	5.0126	12.59 β
SI-URED	0.0094	0.2021	0.2948	12.6772	4.9159	1.98 β
E-STDAC	0.0090	0.1976	0.2581	11.8332	4.3820	2.38 β
I-FDFF	0.0096	0.1975	0.3608	23.3846	4.9785	1.77 β
I-AO-FDFF	0.0050	0.1069	0.1853	10.4184	4.4473	11.15 β
Kalman	0.0052	0.1125	0.1952	8.4625	4.3418	10.09 β
HGD	0.0096	0.2086	0.3097	11.3588	4.6669	2.71 β

Noise type: white, SNR=30dB, h=50ms, and parameters are shown in Table 3.
$\beta = 0.7470$ms. Performance: red<black<blue

To investigate the robustness of the differentiators more clearly, their responses have been analyzed with different SNRs and shown in Figs. 20 to 24. From Fig. 20 (C) to Fig. 24 (C), the Euler method presents the worst responses for SNR<40dB. Comparing the explicit methods in Fig. 20 (A) to Fig. 24(A), it is clear that the E-STD, E-HDD, and E-GHDD show the smallest $\bar{L}_2$ and $\tilde{L}_2$. Note that ALIEN also presents small VAR and THD for SNR>30dB. Moreover, the E-QD shows the worst performances for SNR>30dB, except

for THD in Fig. 24(A).

From Fig. 20 (B) to Fig. 24 (B), it can be seen that the I-QD is the worst implicit method. Furthermore, from Fig. 20 (C) to Fig. 24 (C), the Euler and LF present the worst variations. Among the implicit methods, I-AO-STD presents the best responses for overall SNRs except for the THD where the other implicit schemes are better.

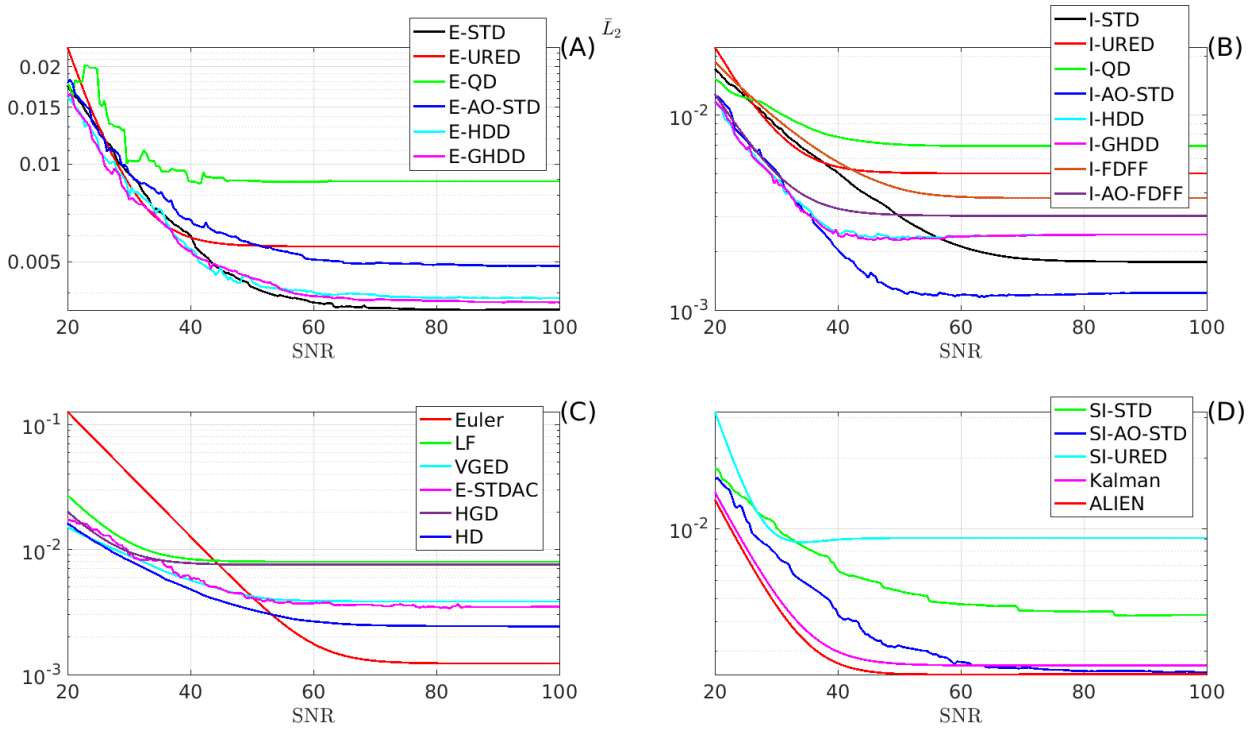

Figure 20: $\bar{L}_2$ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms).

It is seen that the overall behavior of the differentiators remains qualitatively the same in Figs. 20 to 22 for the criteria $\bar{L}_2, \tilde{L}_2$ and L_∞.

It clearly appears from Fig. 20 (A) to Fig. 24 (A), that E-QD is the worst of the explicit "sliding-mode" differentiators. Its performance is equivalent to that of the other explicit schemes, only for small SNRs $\leq$ 25dB. Otherwise, it is much worse. Also note that both VGED and the E-STDAC seem to behave quite similarly for both $\bar{L}_2$ and $\tilde{L}_2$ (Figs. 20 and 21 (C)), but E-STDAC is more accurate for all SNRs (Fig. 22 (C)).

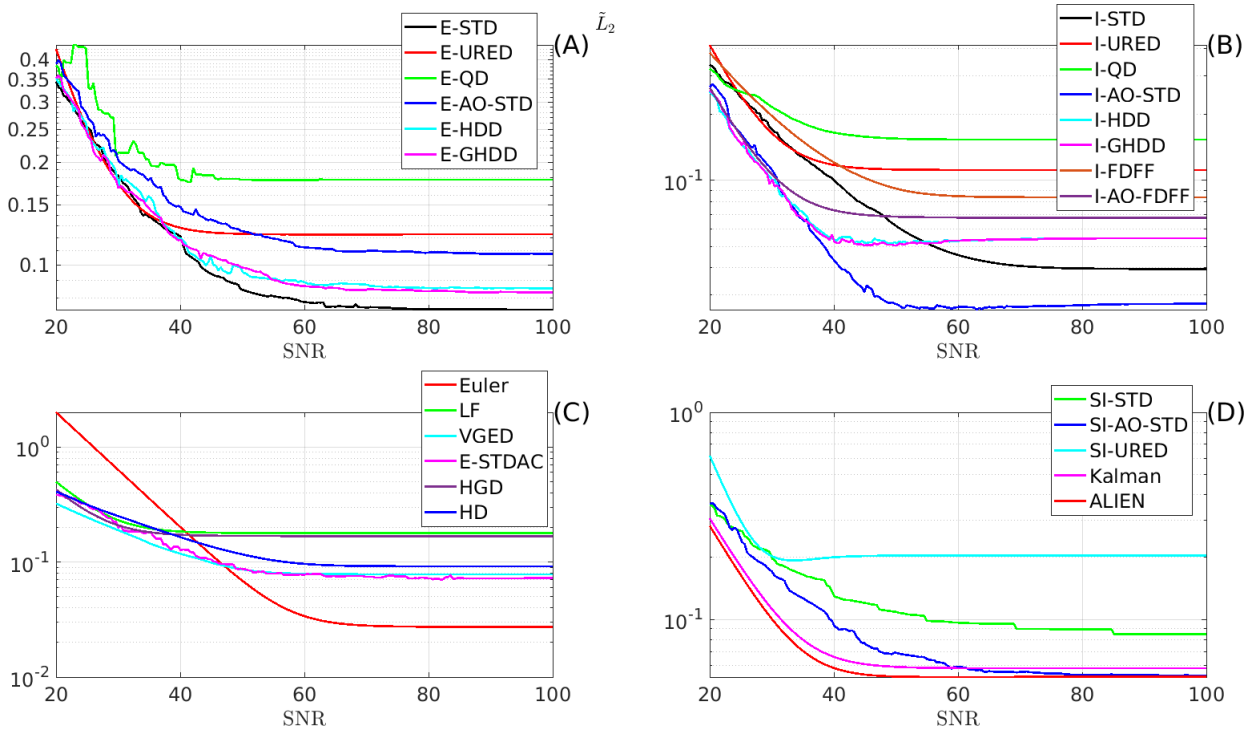

Figure 21: $\tilde{L}_2$ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms).

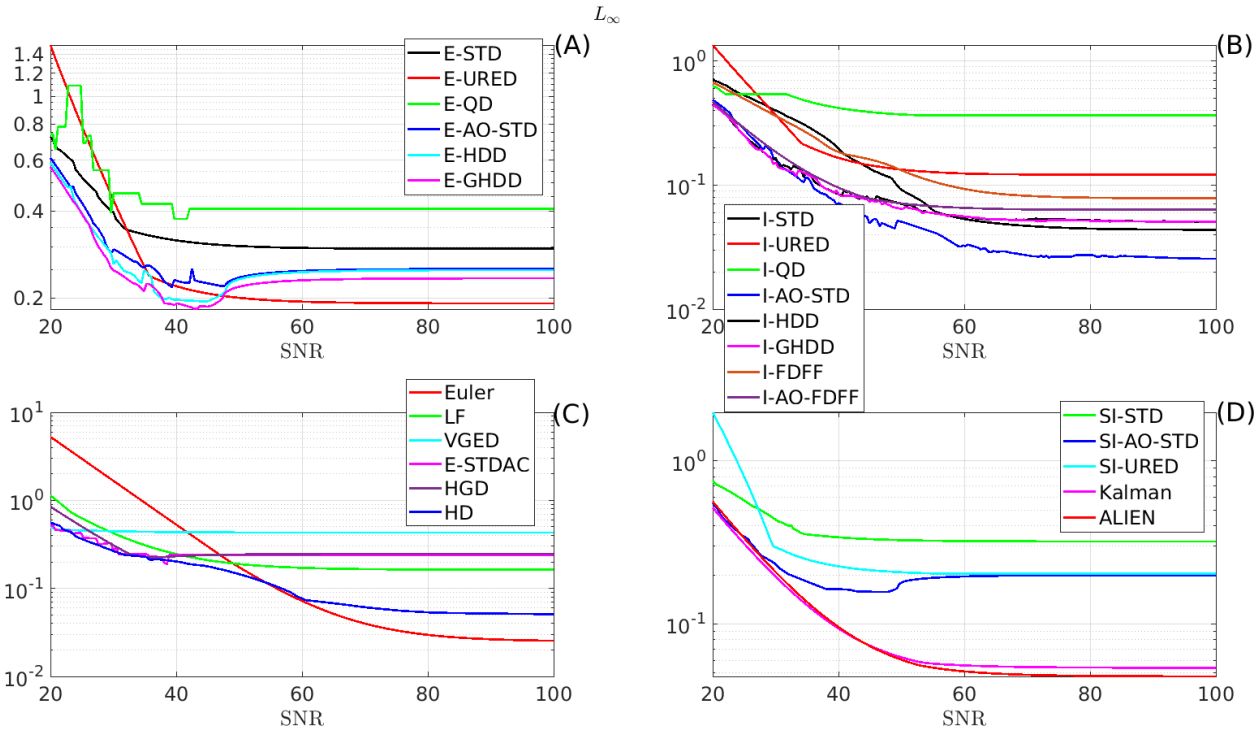

Figure 22: L_∞ for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms).

For small SNR $\leq$ 30dB, VGED and E-STDAC have L_∞ cost comparable to that of implicit schemes and semi-implicit ones (Fig. 22 (C), (B) and (D)). However, they are less accurate for large SNRs.

From Figs. 20 to 24, it appears that I-AO-STD supersedes all other differentiators. Similarly to the previous observations, ALIEN also shows good accuracy (Fig. 22 (D)) since the transient has been been ignored for this simulation. According to Fig. 23, from the chattering point of view, I-AO-STD, I-HDD, I-GHDD and ALIEN perform the best for all SNRs. For explicit schemes, E-AO-STD, E-HDD and E-GHDD supersede all others for SNR $\leq$ 45dB, see Fig. 22 (A). But, implicit counterparts allow smaller VAR in Fig. 22 (B). Also, from the THD point of view in Fig. 24, ALIEN is the best for all SNRs. It should be noted that the HGD behaves almost as the LF. The reason is that for this noisy case, a small gain is obtained for the HGD.

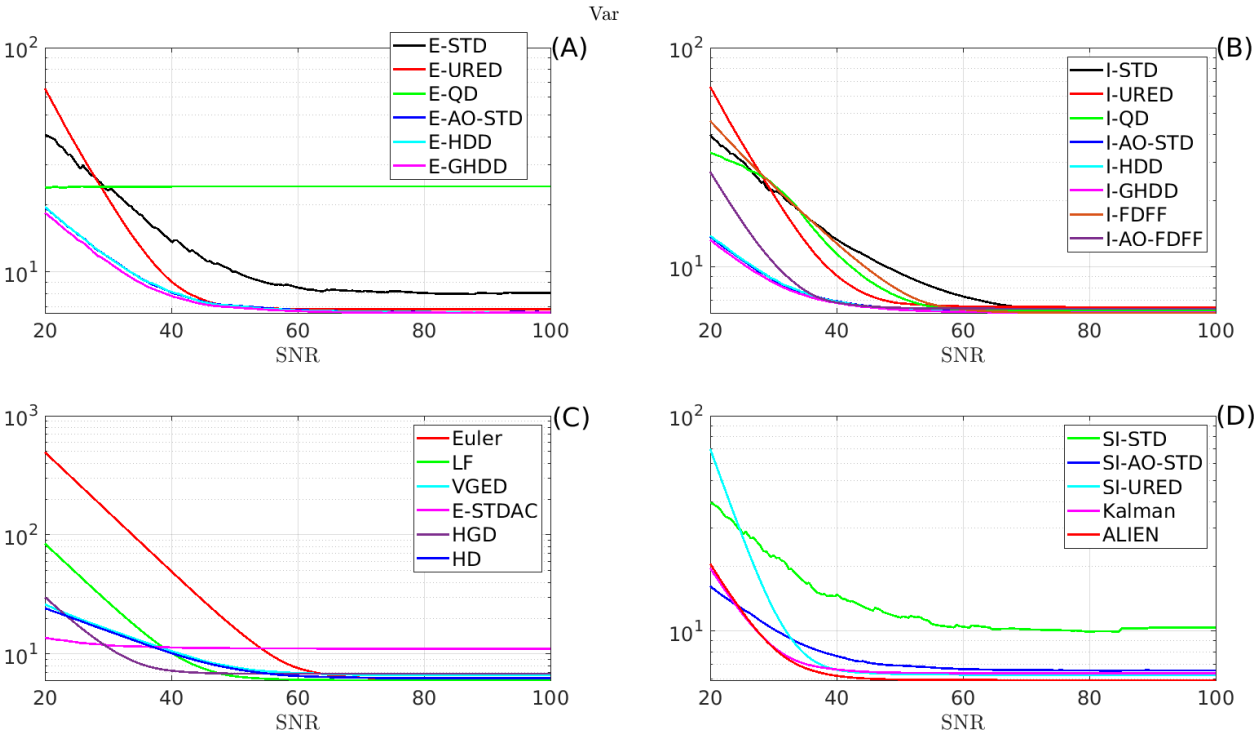

Figure 23: Variation for the first-order differentiation with respect to SNR of the white noise. Parameters are shown in Table 3 ($h = 50$ms).

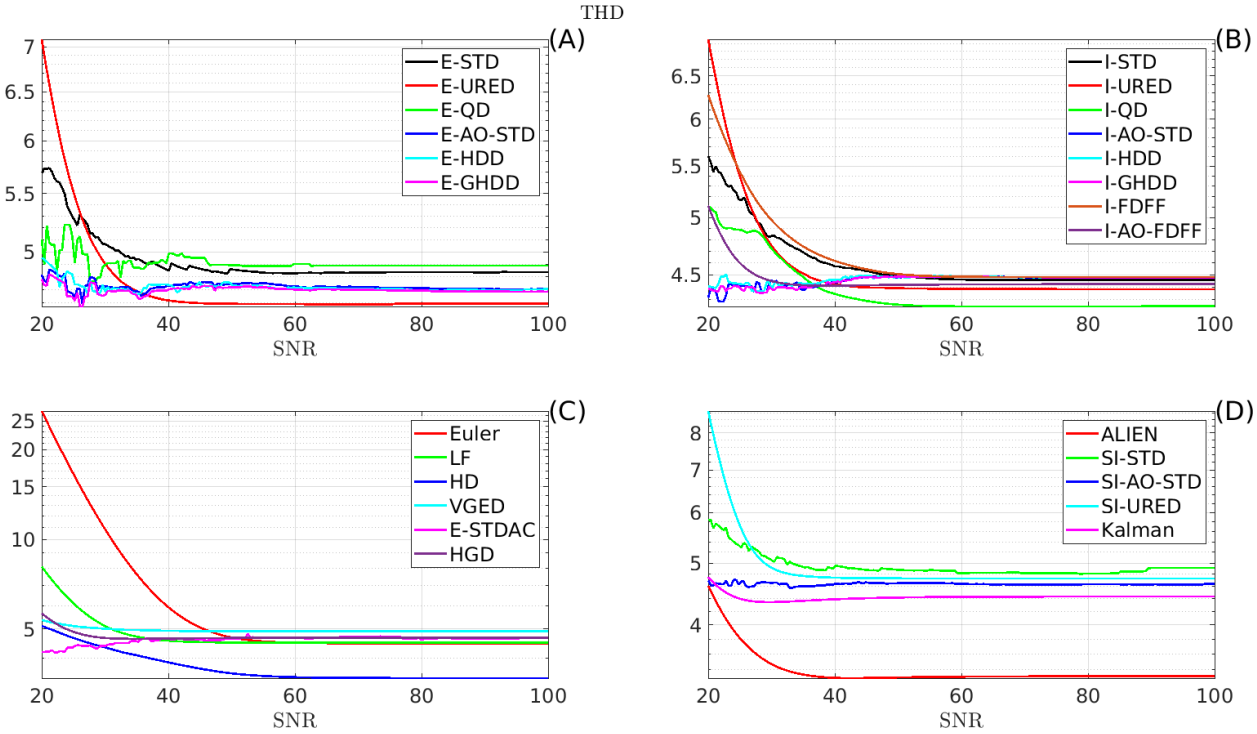

Figure 24: THD for the first-order differentiation concerning SNR. Parameters are shown in Table 3 ($h = 50$ms).

5.2.2 Robustness against sinusoidal noise

To investigate the robustness of the differentiators more deeply, their responses have also been studied in the presence of a sinusoidal noise. The specifications of this noise and the corresponding sub-optimal parameters, which are obtained by the above optimization procedure, are given in Table 6. The results of Table 6 suggest strongly that I-AO-STD, I-HDD, I-GHDD and I-AO-FDFF possess the best performance. LF seems to be the worst one after Euler.

The responses of this experiment are shown in Figs. 25 and 26, and the results are summarized in Table 7. From Table 7, it can be seen that the I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF present the best overall performance in the presence of the sinusoidal noise. According to Table 7, the worst differentiator in the presence of the sinusoidal noise is the Euler method which scores the worst performances for all criteria. Furthermore, LF also presents one of the worst responses. It should be noted that, in general, implicit methods present appropriate responses. Moreover, it can be seen that explicit ones also presented relatively good performances. The reason is that in this noisy condition, performance degradation due to the noise is more dominant than the numerical chattering effect.

Table 6: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2$
Euler (3)	No parameter	480.6846
LF (4)	c=4.9255	152.8754
E-STD (21)	L=0.7540	121.3864
I-STD (Fig. 3)	L=0.7068	117.3041
SI-STD (123)	L=0.7084	130.0751
E-URED (26)	L=0.1193, μ=10.7672	119.9215
I-URED (Fig. 5)	L=0.1068, μ=13.8847	116.7733
E-QD (25)	F=2.7225, α= 0.5349	118.7163
I-QD (60)	F=3.9902, α=0.4315	114.1258
ALIEN (14)	T=0.8155 , κ=1 , μ=5	69.4016
HD (40)*	r=2.1642	172.7417
E-AO-STD (22)**	L=4.5499	128.8121
I-AO-STD (Fig. 7) **	L=3.4498	68.9626
SI-AO-STD (122)**	L=5.0980	96.7735
E-HDD (24)**	L=4.2446	112.5950
E-GHDD (27)**	L=4.5529	112.0893
I-HDD (Fig. 9)**	L=3.5819	59.8965
I-GHDD (Fig. 10)**	L=5.4279	60.5955
VGED (35)	μ=4.7459, τ=5.5715, ω_c=5.2918, q=0.0054	117.0100
SI-URED (139)	L=0.0494, μ=97.4981	102.2382
E-STDAC (10)	α=0.7878, ϵ=0.0206	128.2747
I-FDFF (Fig. 12)	ω_s=79.7566, ω_f=1.6261, ρ=85.3648, γ=0.0038	91.1546
I-AO-FDFF (Fig. 12)	F=39.6543, ϵ=11.3619, ω_s=2.2867	65.3488
	ω_f=83.2350, α_1=136.6326, ρ=101.3983	
Kalman (Section 3.5)	R=0.0016	53.4839
HGD (16)	L=3.1287	104.8582
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]). ** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation (output is z_1) (see (7)).		
Input signal: sin(t), Output: first-order differentiation, Performance: red<black<blue Noise type: sinusoidal, noise: 0.05sin(20t), h = 50ms		

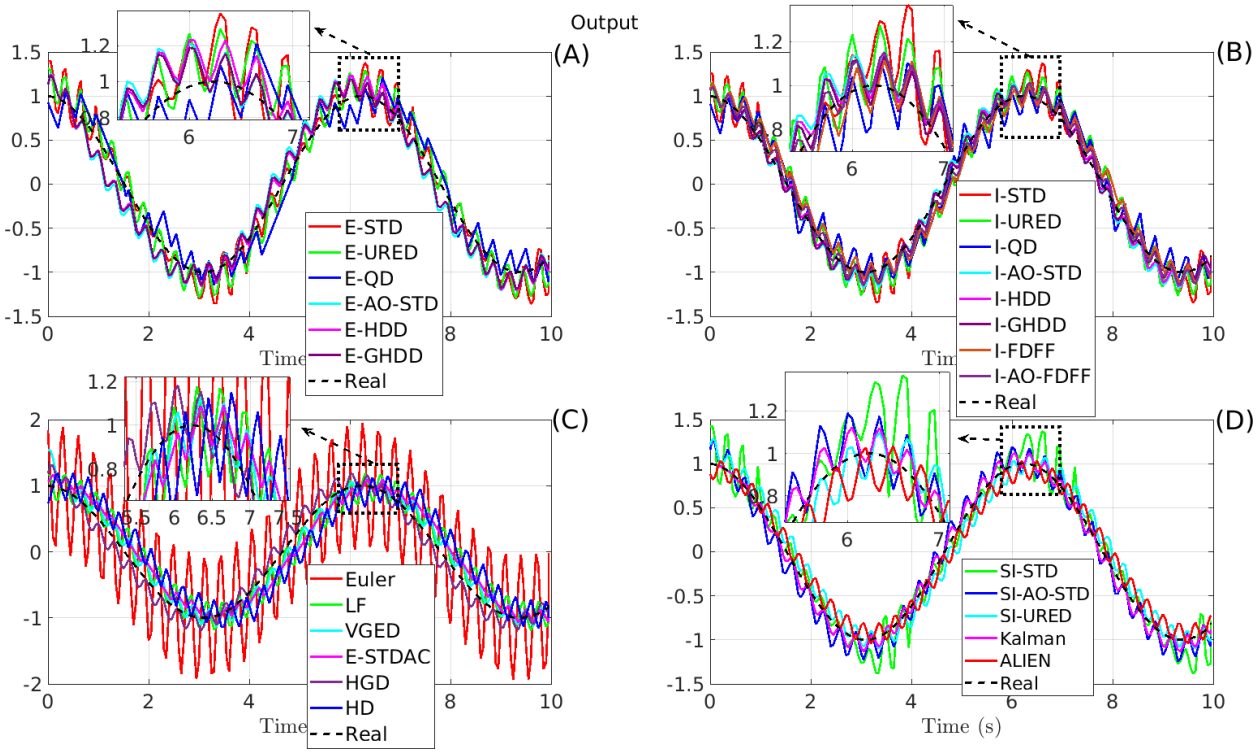

Figure 25: First-order differentiation under sinusoidal noise 0.05sin(20t). Parameters are shown in Table 6 ($h = 50$ms).

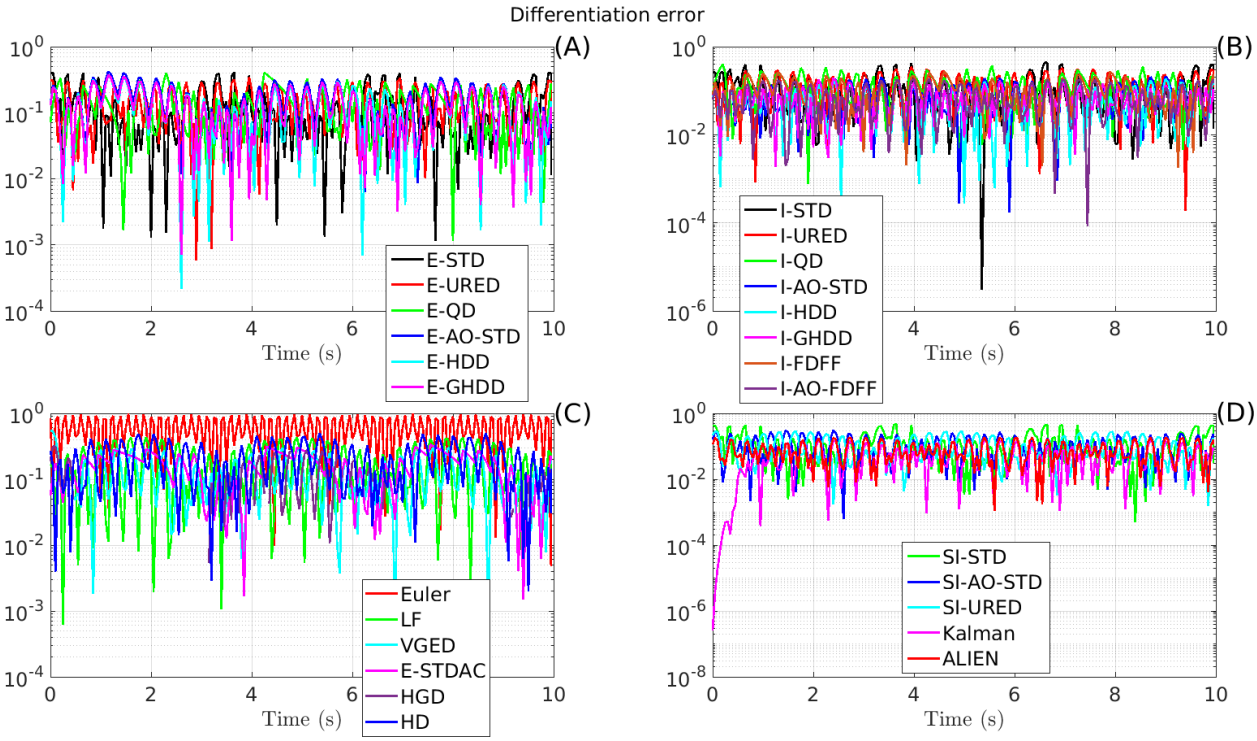

Figure 26: Error of the first-order differentiation under sinusoidal noise 0.05sin(20t). Parameters are shown in Table 6 ($h = 50$ms).

Table 7: Results under a sinusoidal noise. Conditions are mentioned in Table 6.

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0481	0.9835	0.9805	117.7242	5.6183	1.00 β
LF	0.0153	0.3279	0.4341	26.5032	4.7383	1.29 β
E-STD	0.0121	0.2511	0.4111	23.1351	4.9361	1.81 β
I-STD	0.0117	0.2461	0.4370	20.6398	4.7390	3.59 β
SI-STD	0.0130	0.2693	0.4793	22.7952	5.0230	1.64 β
E-URED	0.0120	0.2523	0.3431	23.6801	4.6893	2.06 β
I-URED	0.0117	0.2460	0.3462	23.4952	4.5392	80.76 β
E-QD	0.0119	0.2562	0.4036	18.9317	4.7212	2.61 β
I-QD	0.0114	0.2445	0.3994	20.1828	4.6024	4.69 β
ALIEN	0.0069	0.1457	0.1873	14.4241	3.5421	30.95 β
HD	0.0173	0.3757	0.5023	26.6097	3.6381	8.18 β
E-AO-STD	0.0129	0.2826	0.4230	15.6376	4.8603	3.60 β
I-AO-STD	0.0069	0.1472	0.2094	12.2203	4.5656	18.21 β
SI-AO-STD	0.0097	0.2043	0.3078	19.2835	4.9150	3.48 β
E-HDD	0.0113	0.2457	0.3689	15.1775	4.8155	3.66 β
E-GHDD	0.0112	0.2450	0.3738	14.7486	4.8450	4.88 β
I-HDD	0.0060	0.1254	0.1846	12.4579	4.5868	17.64 β
I-GHDD	0.0061	0.1262	0.1923	14.0411	4.5691	21.47 β
VGED	0.0117	0.2509	0.5394	16.3756	5.1918	8.38 β
SI-URED	0.0102	0.2208	0.2920	13.5724	4.8437	2.78 β
E-STDAC	0.0128	0.2836	0.3350	12.0935	4.7146	2.91 β
I-FDFF	0.0091	0.1971	0.2695	14.7789	4.6494	3.31 β
I-AO-FDFF	0.0065	0.1386	0.1840	12.1946	4.5383	12.72 β
Kalman	0.0053	0.1134	0.1570	11.0079	4.4905	4.81 β
HGD	0.0105	0.2271	0.3176	16.1519	4.8171	1.09 β

Noise type: sinusoidal, noise: 0.05sin(20t), h=50ms, and parameters are shown in Table 6. $\beta = 0.1918$ms. Performance: red<black<blue

Robustness of the differentiators is also investigated with respect to sinusoidal noise by changing the frequency of the noise. The performances of the differentiators with respect to the noise frequency are shown in Figs. 27 to 31. According to these figures, it can be seen that for $\omega > 10$ Rad/s, the performances of all algorithms are almost constant for different noise frequencies, except for the Euler method. In fact, by increasing the noise frequency, the performance of the Euler method will be deteriorated almost all

algorithms. Almost all algorithms' response show a peak of low frequency and then improvement of the criteria for larger frequencies. Once again, the qualitatively behavior of the differentiators is almost constant for the three criteria $\bar{L}_2, \tilde{L}_2, L_\infty$ in Figs. 27 to 29.

According to Figs. 27 to 31, it can be seen that implicit, semi-implicit, ALIEN and Kalman methods presented the best responses. Among them, I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF are the best ones. Furthermore, SI-AO-STD and SI-URED also present appropriate responses.

Considering the chattering (VAR in Fig. 30), it is worth noting that some differentiators have an almost constant VAR for frequencies ≥ 5 rad/s (SI-AO-STD, SI-URED I-AO-STD, I-HDD, I-GHDD, E-QD) and others have increasing chattering with increasing noise frequency (I-STD, E-URED, SI-STD,E-STD). Note that by increasing the frequency, the chattering of the ALIEN will be decreased. The reason is that it is based on integration which allows to attenuate the high frequency oscillations of the noise more effectively, compared to the other schemes.

Concerning the harmonic attenuation, ALIEN has the best spectrum response with the smallest THD (Fig. 31 (D)), better than I-AO-STD. I-QD also shows very good spectrum response in Fig. 31. Moreover, there is not a significant difference between explicit and implicit versions concerning their THD in Fig. 31 (A) and (B), though implicit ones slightly supersede explicit ones.

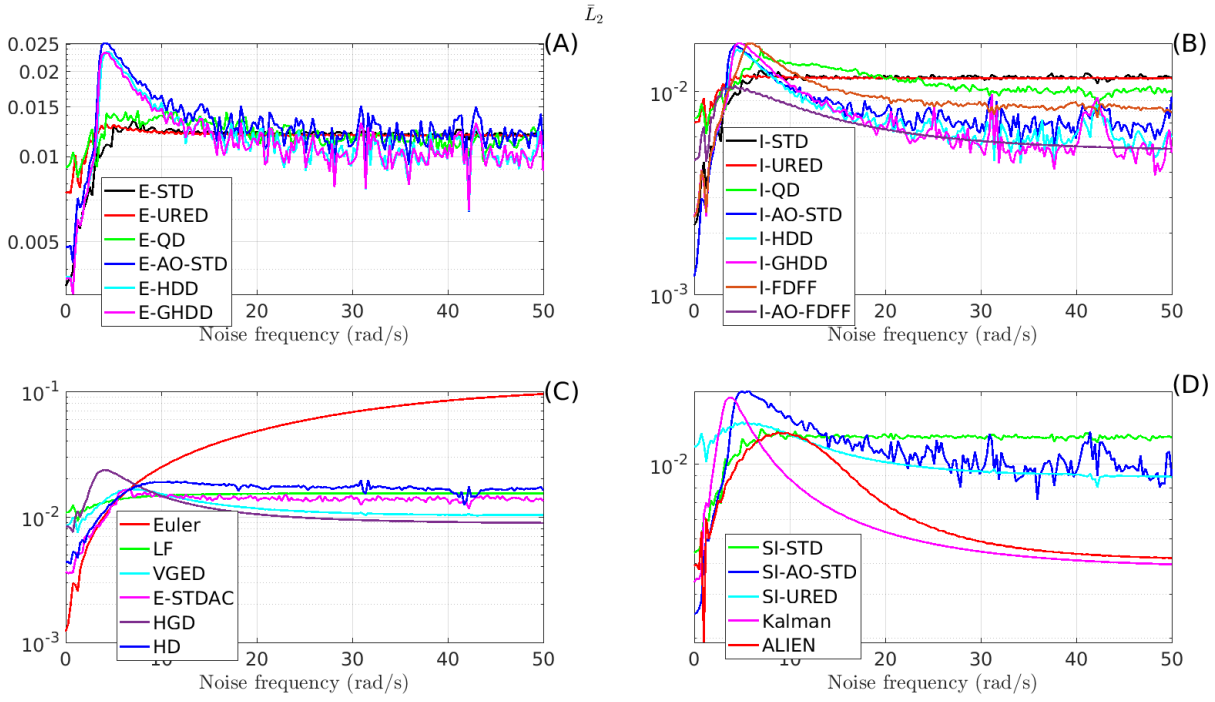

Figure 27: $\bar{L}_2$ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms).

We see from Fig. 27 (A) to Fig. 31 (A) that contrarily to the case of white noise, this time E-QD has a similar performance to the rest of the explicit differentiators. It will be inferred next that where the noise

amplitude increases, then E-QD can supersede the other explicit schemes (Fig. 32 (A) to Fig. 36 (A)).

While both HGD and LF have linear structures, the HGD shows slightly better responses for $\omega > 10\,\mathrm{rad/s}$. On the other hand, the LF shows a better behavior for smaller frequencies of the noise ($3 < \omega < 10\,\mathrm{rad/s}$).

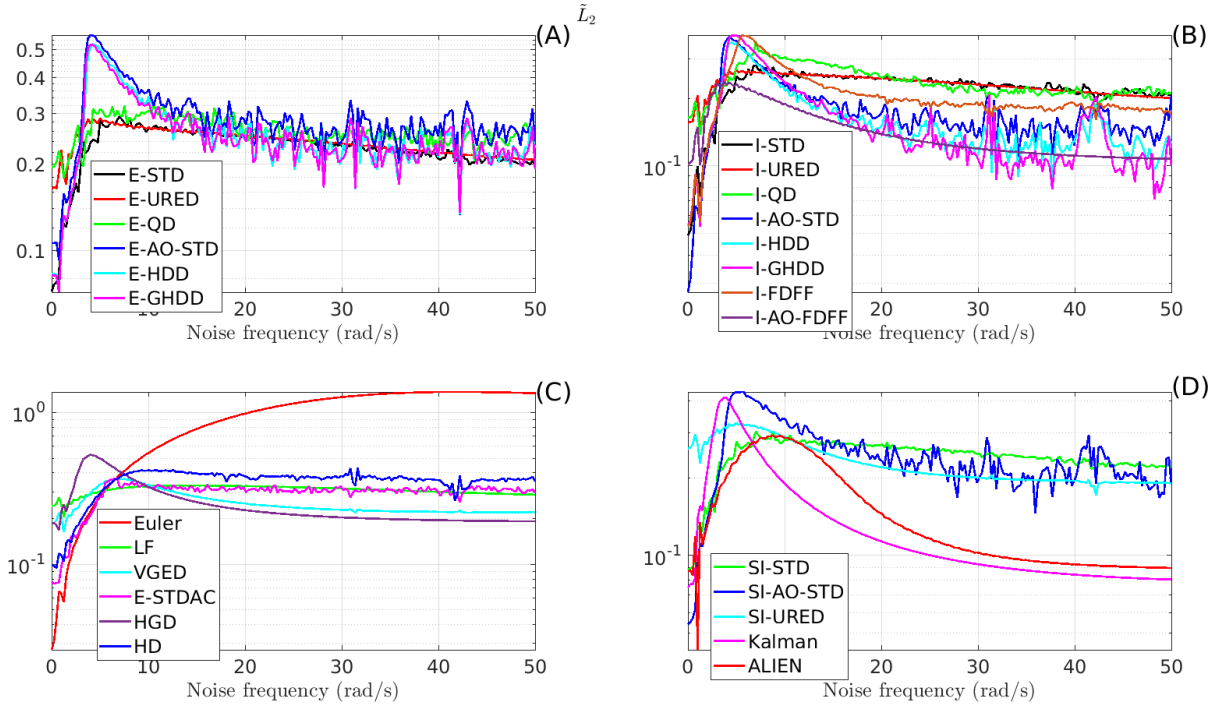

Figure 28: $\tilde{L}_2$ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50\mathrm{ms}$).

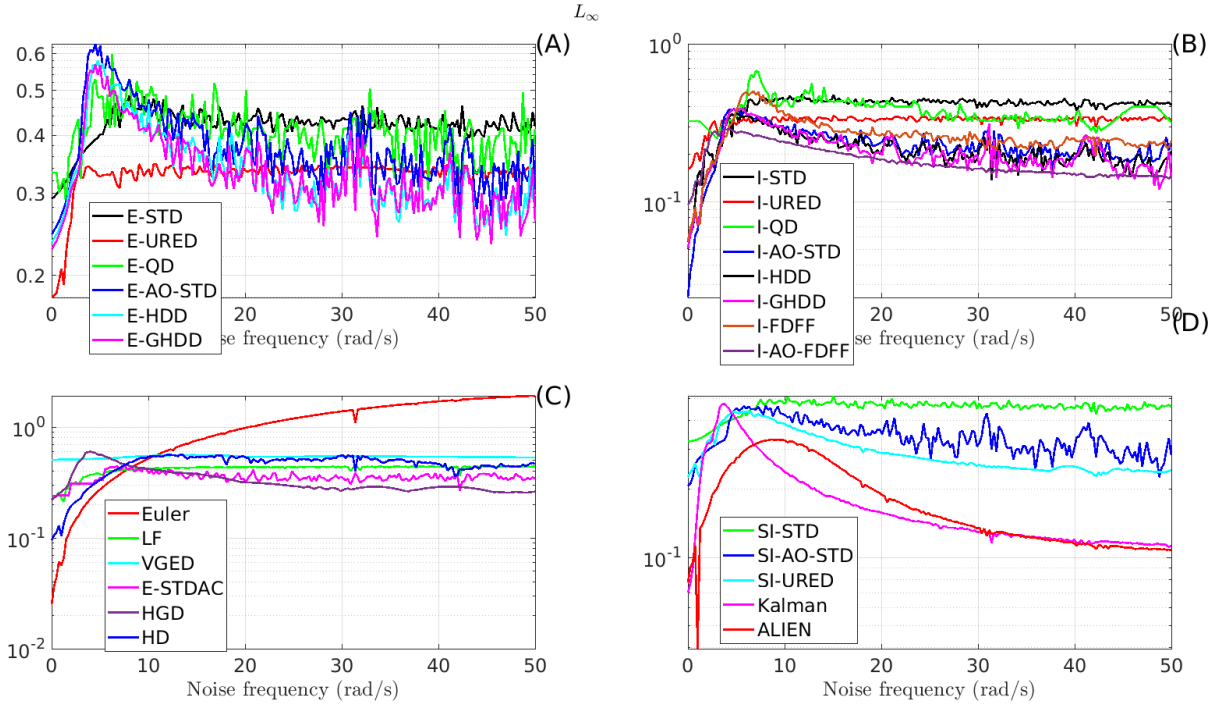

Figure 29: L_∞ for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms).

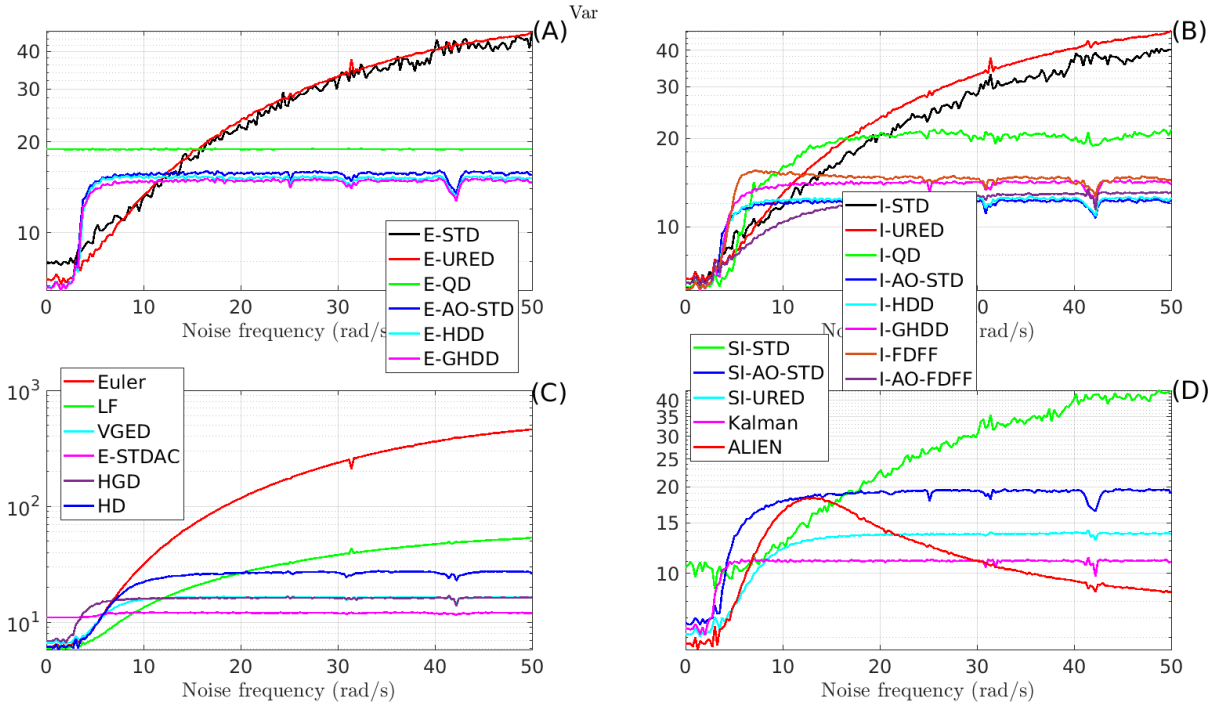

Figure 30: Variation for the first-order differentiation with respect to noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms).

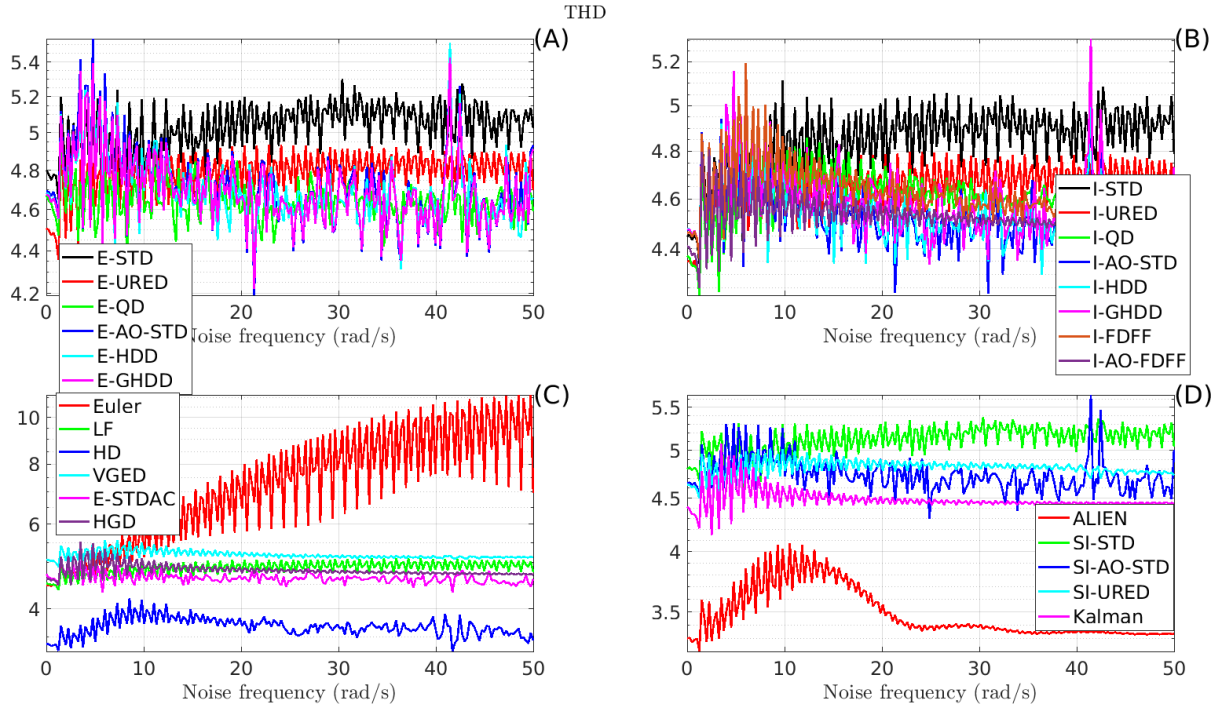

Figure 31: THD for the first-order differentiation concerning noise frequency. Input noise: $0.05\sin(\omega t)$, ω is the noise frequency. Parameters are shown in Table 6 ($h = 50$ms).

The performance of the differentiators is now studied by changing the amplitude of the input noise in Figs. 32 to 36, while the noise frequency remains constant ($\omega = 20\,\mathrm{rad}/s$). These figures show that increasing the amplitude of the noise deteriorates all the performances. Figs. 32, 33, and 34 show that I-AO-STD, I-HDD, I-GHDD, I-QD and VGED present the best responses for large noise amplitudes. Among them, VGED and HD seems to be the best one for large amplitudes. However, for small noise amplitudes, the I-AO-STD, I-HDD, I-GHDD, ALIEN and Kalman are the best ones for almost all the criteria. Note that the SI-AO-STD also shows good responses for amplitudes larger than 0.1. Moreover, E-QD presents one of the best responses in the variation (chattering) point of view as shown in Fig. 35 (A). Once again, the results in Figs. 32 to 34 prove that the overall qualitative performances do not vary much for the three criteria $\bar{L}_2, \tilde{L}_2, L_\infty$.

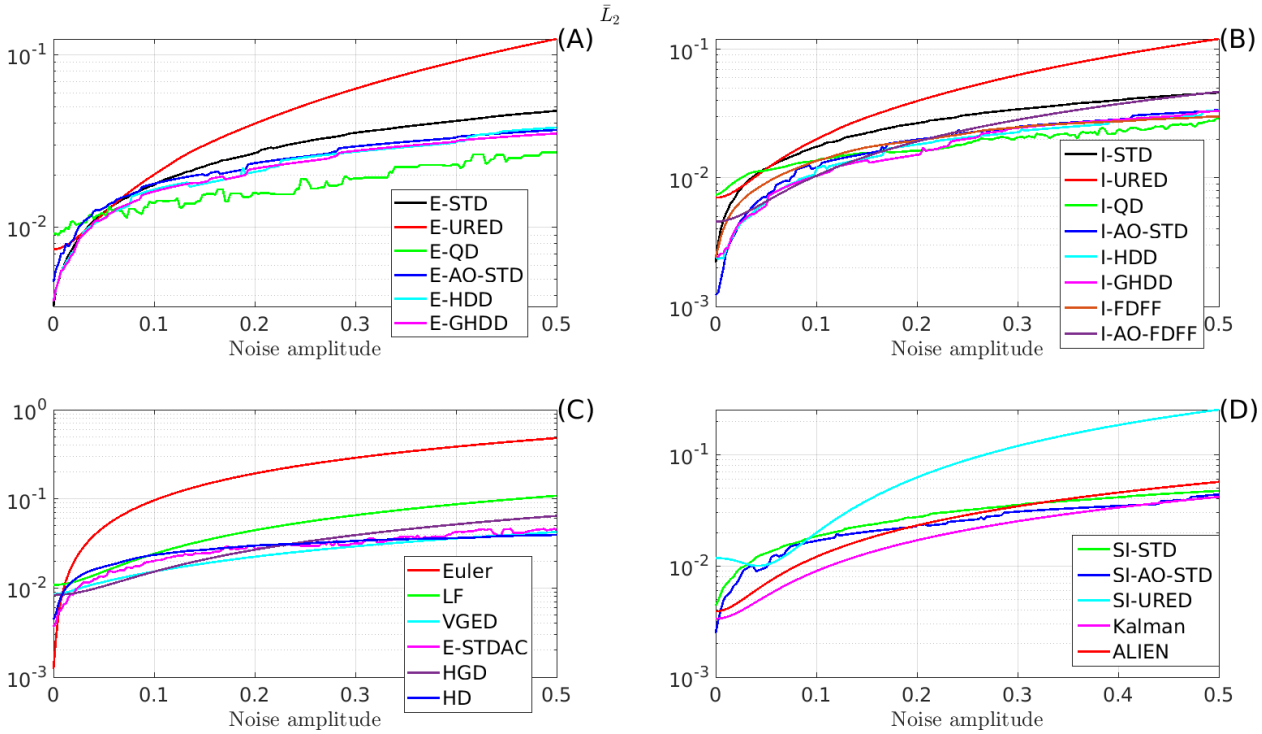

Figure 32: $\bar{L}_2$ for the first-order differentiation with respect to noise amplitude. Input noise: A $\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms).

It is also visible on Figs. 32 to 36 (A) and (B) that E-AO-STD, E-HDD, E-GHDD and I-AO-STD, I-HDD, I-GHDD, possess very close performances (their curves are almost the same). Also, comparing the results of the HGD and the LF, one concludes that the HGD always behaves better than the LF.

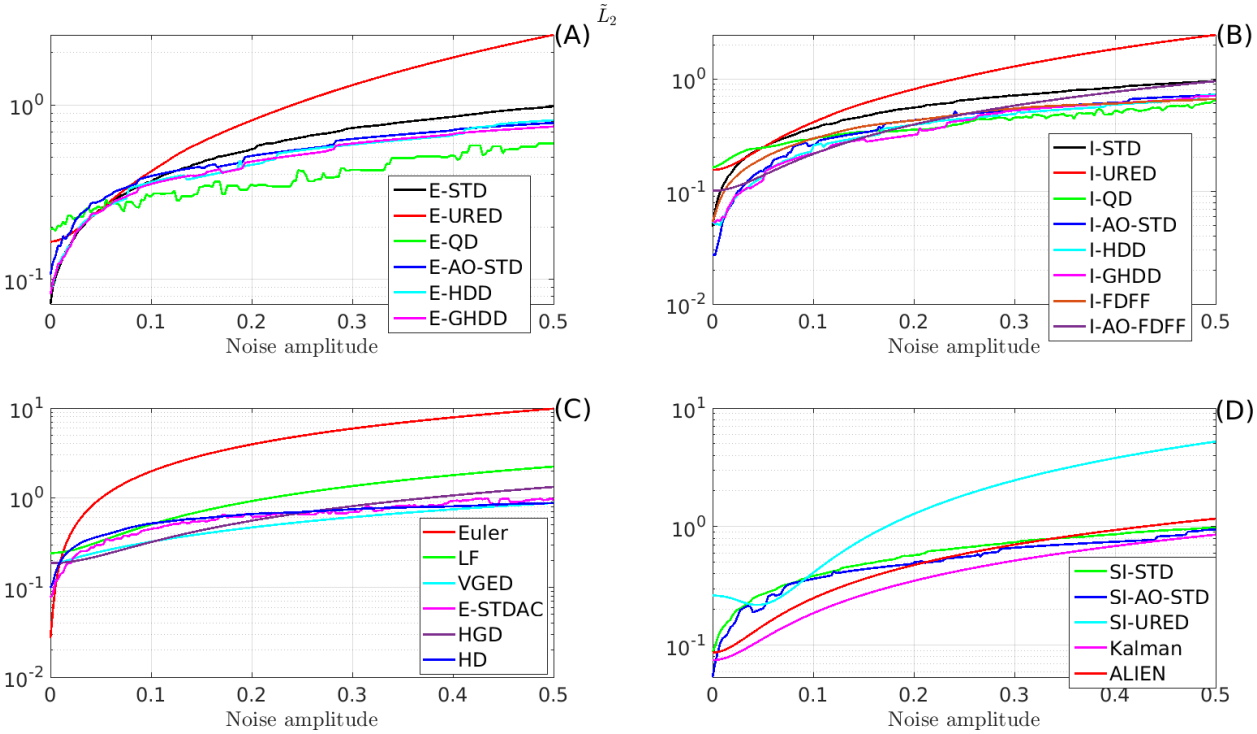

Figure 33: $\tilde{L}_2$ for the first-order differentiation with respect to noise amplitude. Input noise: A $\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms).

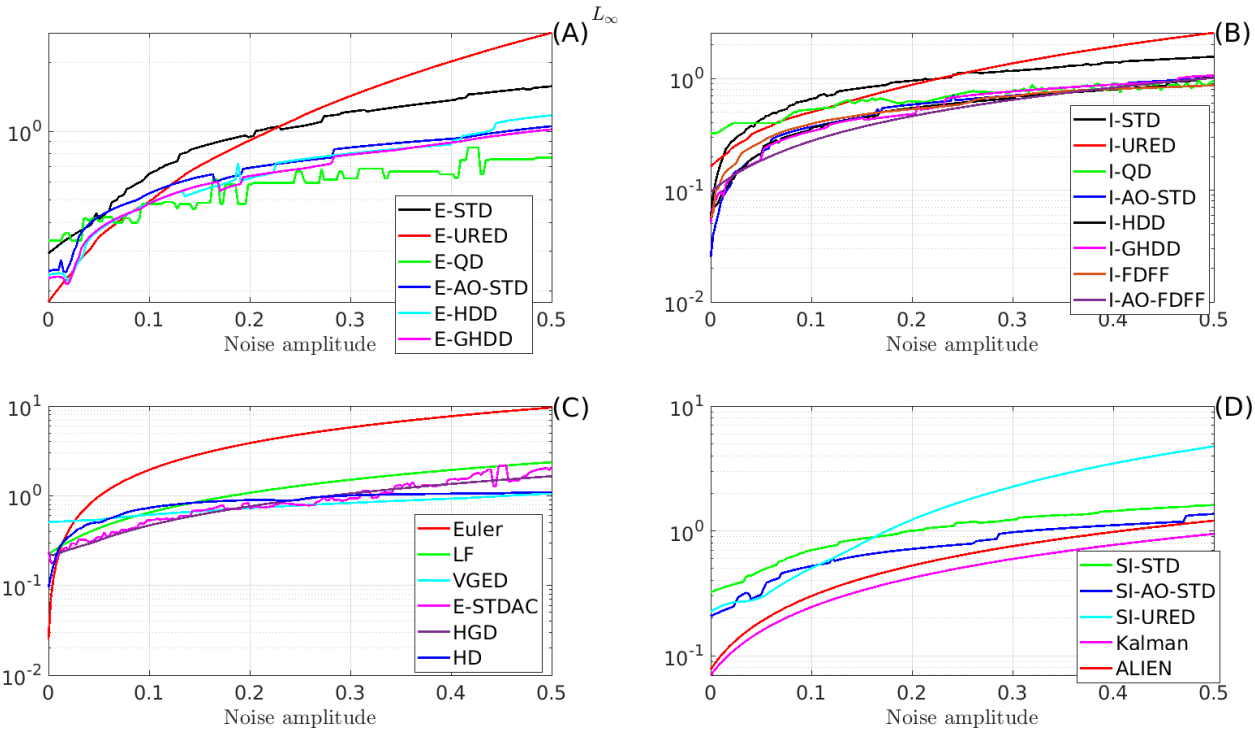

Figure 34: L_∞ for the first-order differentiation with respect to noise amplitude. Input noise: A $\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms).

As alluded to above, the performance of QD improves a lot (compared with other differentiators) when the noise amplitude increases, see Figs. 32 to 34 (A) and (B).

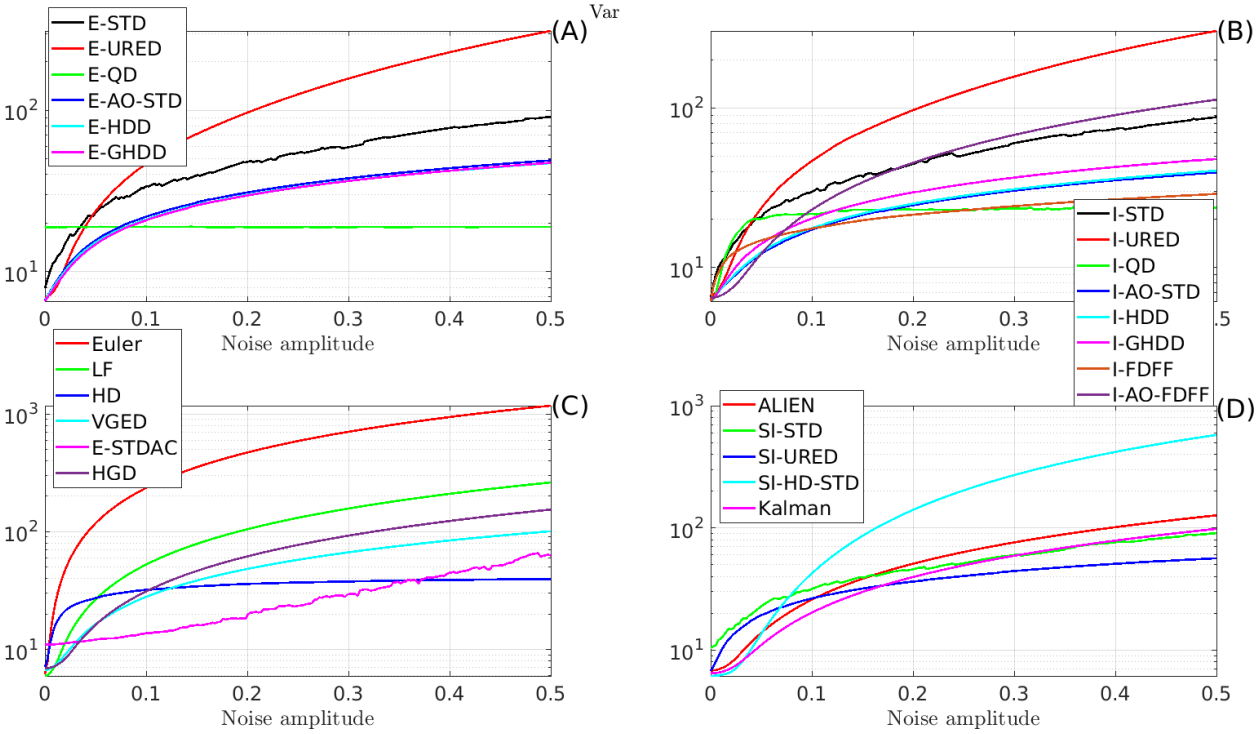

Figure 35: Variation for the first-order differentiation with respect to noise amplitude. Input noise: A $\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms).

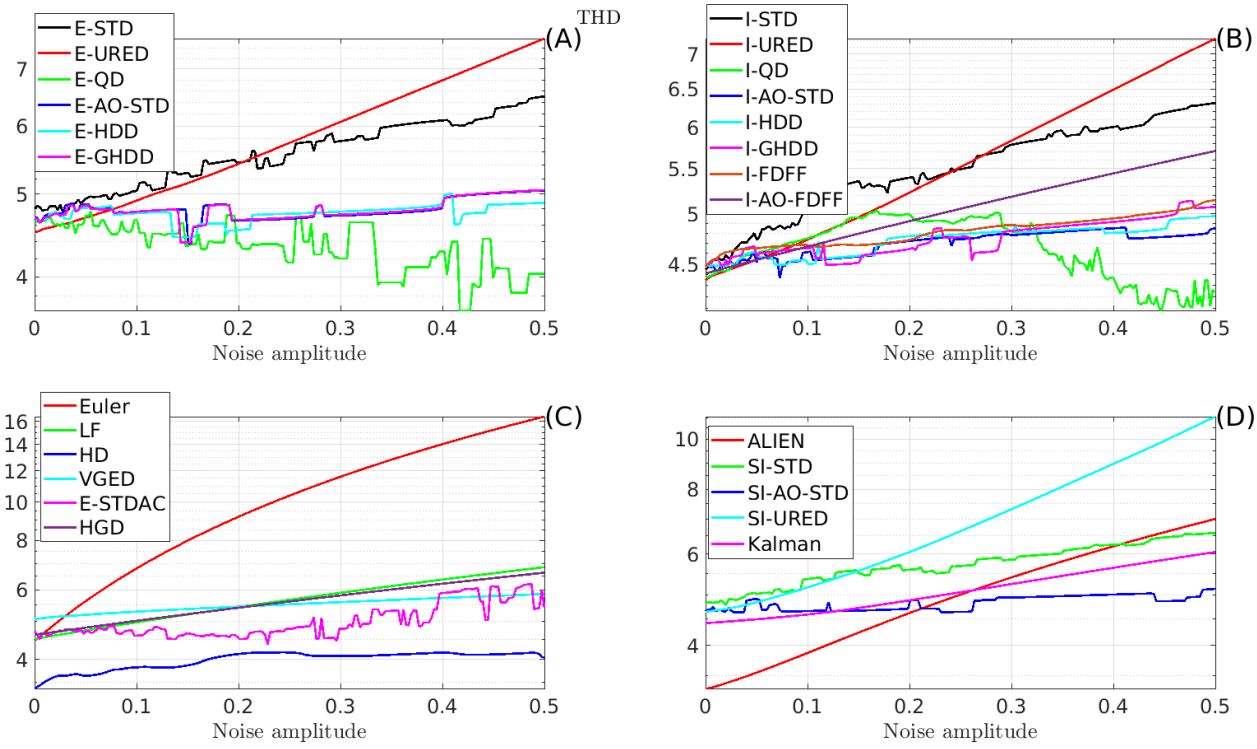

Figure 36: THD for the first-order differentiation with respect to noise amplitude. Input noise: A $\sin(20t)$, A is the noise amplitude. Parameters are shown in Table 6 ($h = 50$ms).

From Figs. 32 and 33 (A), (B) and (D) one sees that I-AO-STD, I-HDD, I-GHDD, and E-QD present better $\bar{L}_2$ and $\tilde{L}_2$ than others for large noise amplitudes (≥ 0.2).

5.2.3 Robustness against bell-shaped noise

So far, the robustness of the differentiators have been investigated for white and sinusoidal types of noises in Sections 5.2.1 and 5.2.2, respectively. The spectrum analysis of sinusoidal and white noises, for $h = 50$ms, are calculated using the FFT and shown in Fig. 37 (A) and (C), respectively. As it expected, while the sinusoidal noise has only one frequency component, the power of the white noise is almost constant for the frequency range.

Sinusoidal noise enables to investigate the behavior of the differentiators for a single frequency noise. On the other hand, white noise can be used to study the effect of the noise for all frequency components. However, from the practical point of view, these types of noises may not be realistic enough. To address this ambiguity, another type of noise called bell-shaped or Gaussian-like noise, which is shown in Fig. 37 (F) (E), is considered in this section. The bell-shaped noise is generated by putting a band-pass filter on a white noise with SNR=30dB.

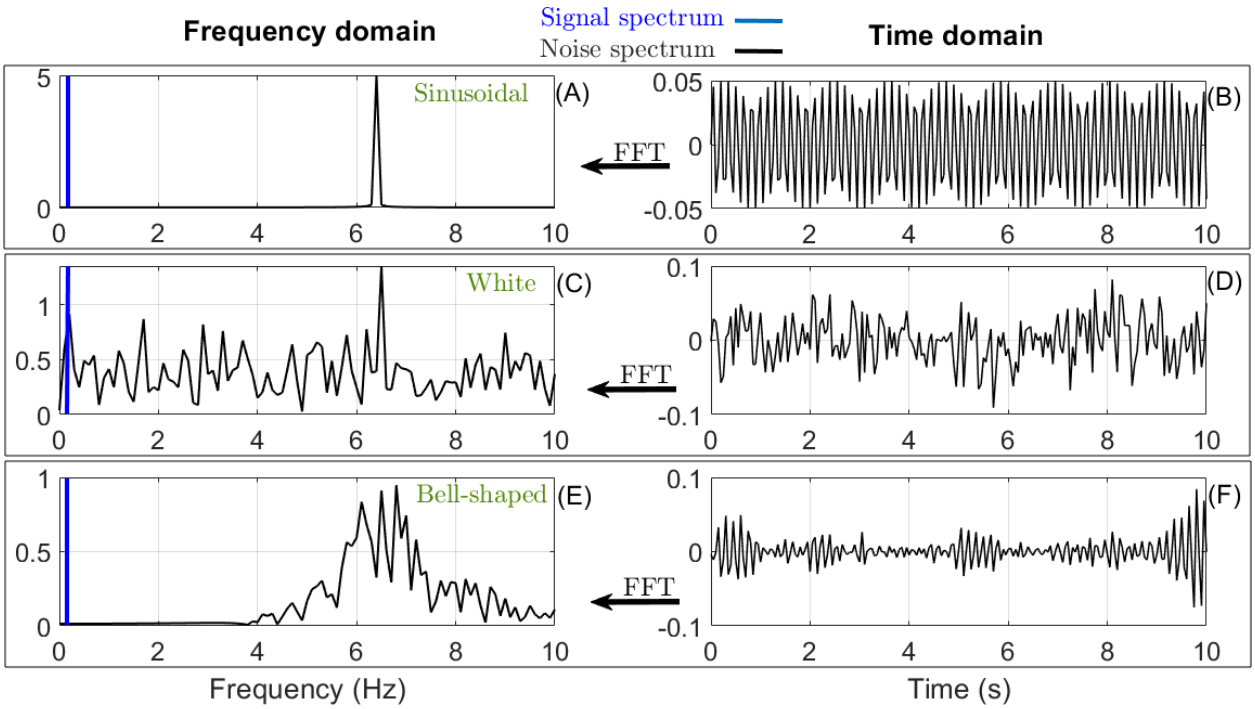

Figure 37: Spectrum analysis of different types of noises. (A, B): sinusoidal noise, (C, D): white noise, and (E, F): bell-shaped noise.

The results of the gain-tuning optimization process are shown in Table 8. Once again and coherently with foregoing case, I-AO-STD, I-HDD and I-GHDD possess the smallest objective functions J.

Table 8: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2$
Euler (3)	No parameter	203.7077
LF (4)	c=11.9159	74.8117
E-STD (21)	L=0.7539	61.9498
I-STD (Fig. 3)	L=0.7674	54.0818
SI-STD (123)	L=0.7247	72.0530
E-URED (26)	L=0.0461, μ=51.8031	53.1278
I-URED (Fig. 5)	L=0.0466, μ=75.5819	45.6699
E-QD (25)	F=3.2690, α= 0.5755	88.9084
I-QD (60)	F=7.6409, α=0.1318	62.2972
ALIEN (14)	T=0.5510 , κ=1 , μ=3	32.0664
HD (40)*	r=1.6127	49.8895
E-AO-STD (22)**	L=4.9825	60.5120
I-AO-STD (Fig. 7) **	L=2.8225	15.2922
SI-AO-STD (122)**	L=3.9572	40.6361
E-HDD (24)**	L=2.5474	48.5573
E-GHDD (27)**	L=3.1314	47.5906
I-HDD (Fig. 9)**	L=3.0027	20.2940
I-GHDD (Fig. 10)**	L=1.9054	19.9382
VGED (35)	μ=3.0700, τ=7.9892, ω_c=11.2710, q=0.4577	62.0122
SI-URED (139)	L=2.4712, μ=35.4051	65.4048
E-STDAC (10)	α=0.2454, ϵ=0.0003	59.8877
I-FDFF (Fig. 12)	ω_s=30.5203, ω_f=0.2184, ρ=32.5707, γ=0.0346	50.4773
I-AO-FDFF (Fig. 12)	F=95.3553, ϵ=40.7603, ω_s=2.9696	24.7029
	ω_f=137.3185, α_1=175.1103, ρ=109.8355	
Kalman (Section 3.5)	$R = 1.6341 \times 10^3$	14.2326
HGD (16)	L=4.5057	68.4768
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]). ** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation (output is z_1) (see (7)).		
Input signal: $\sin(t)$, Output: first-order differentiation, Performance: red<black<blue Noise type: Bell-shaped (Fig. 37 (E) and (F)), h = 50ms		

It appears from Table 8 and gains tuning with $\bar{L}_2$ cost, that four algorithms I-AO-STD, I-HDD, I-GHDD,

and I-AO-FDFF, possess the best levels of performance, while the Euler is catastrophic. Comparing LF and the HGD, it can be seen that the HGD provides a slightly smaller $\bar{L}_2$.

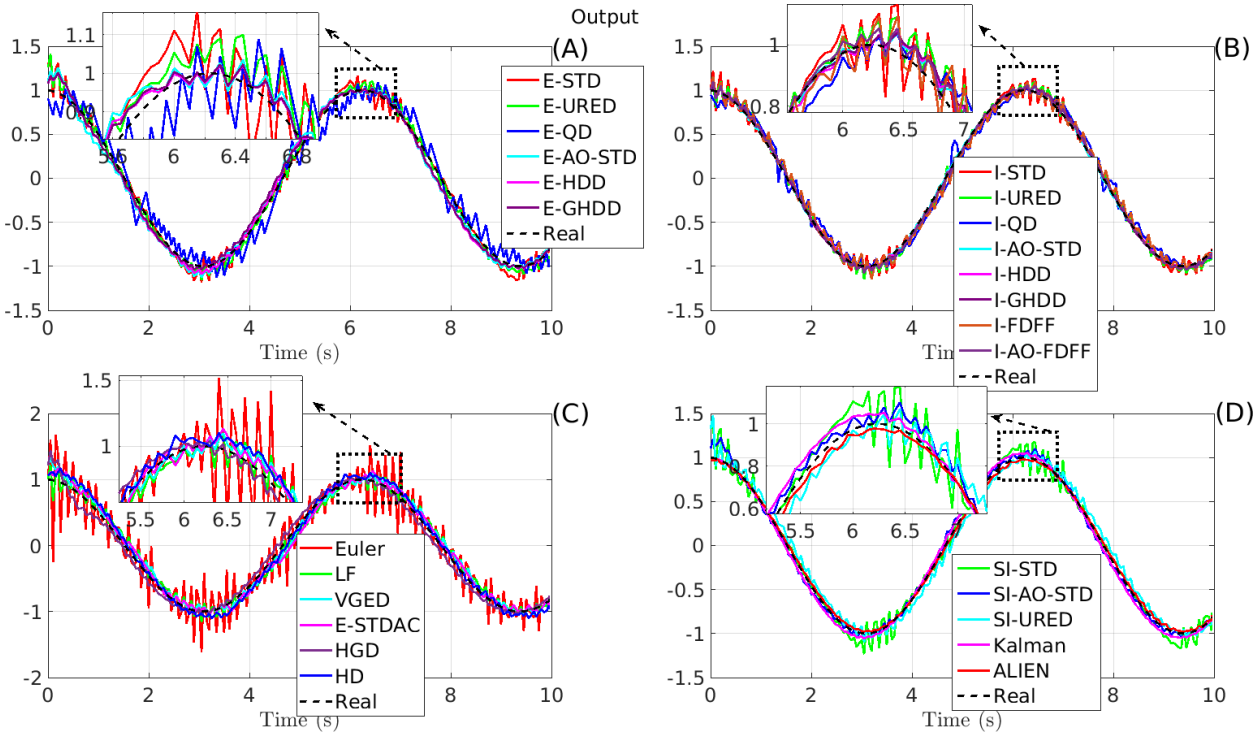

Figure 38: First-order differentiation under a bell-shaped noise ($h = 50$ms).

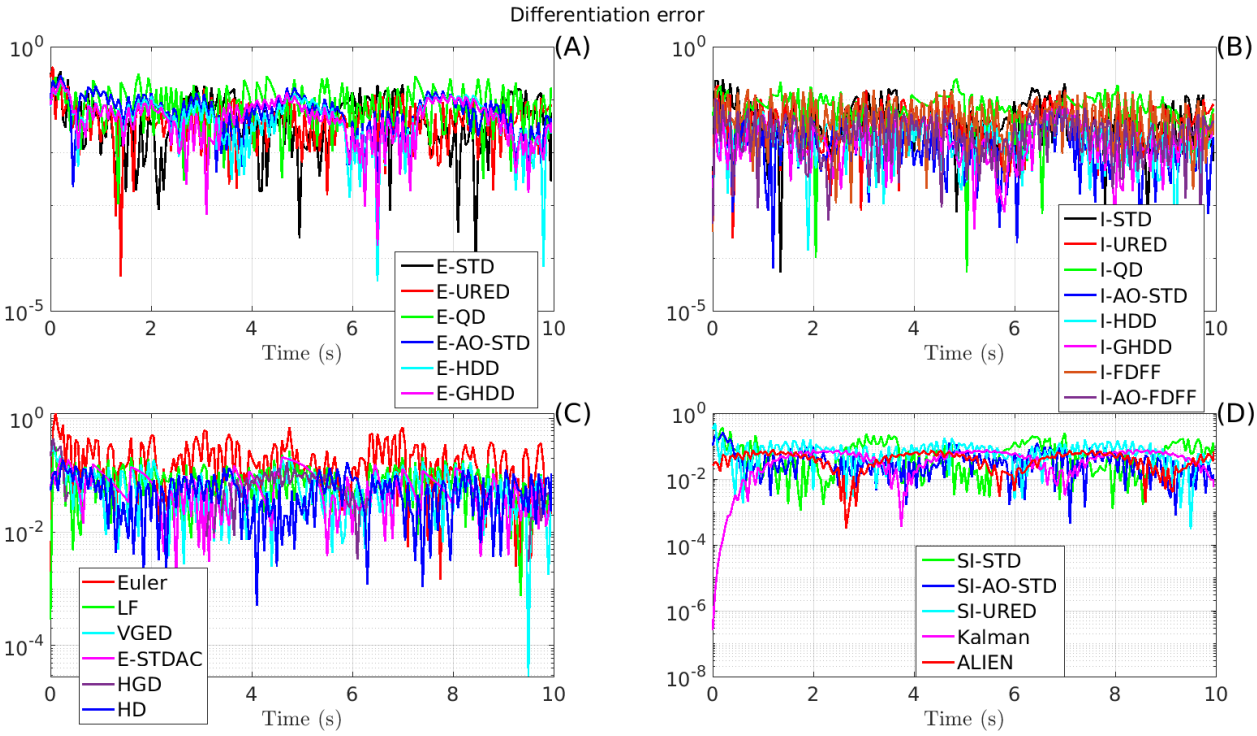

Figure 39: First-order differentiation error under a bell-shaped noise ($h = 50$ms).

Table 9: Results under a bell-shaped noise.

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0204	0.3152	1.2380	80.3500	6.8757	1.00 β
LF	0.0075	0.1441	0.3152	21.0743	4.8234	1.32 β
E-STD	0.0062	0.1143	0.3793	17.6677	4.9242	1.90 β
I-STD	0.0054	0.0920	0.2398	18.9912	4.7271	4.33 β
SI-STD	0.0072	0.1297	0.4068	19.9626	5.0887	1.80 β
E-URED	0.0053	0.1027	0.4172	14.5767	4.8975	2.12 β
I-URED	0.0046	0.0836	0.1862	15.2755	4.6275	16.07 β
E-QD	0.0089	0.1823	0.3070	23.8080	4.7774	2.51 β
I-QD	0.0062	0.1282	0.2428	12.7928	4.5884	6.39 β
ALIEN	0.0032	0.0706	0.0771	6.2803	4.1391	8.50 β
HD	0.0050	0.1059	0.1855	10.5484	3.9449	4.36 β
E-AO-STD	0.0061	0.1323	0.2874	8.7799	4.6880	3.68 β
I-AO-STD	0.0015	0.0300	0.0664	7.2491	4.4876	18.06 β
SI-AO-STD	0.0041	0.0859	0.2567	9.1864	4.6484	3.81 β
E-HDD	0.0049	0.1067	0.2236	7.7065	4.5420	4.01 β
E-GHDD	0.0048	0.1044	0.2304	7.7114	4.6012	5.22 β
I-HDD	0.0020	0.0420	0.0791	7.3112	4.5028	17.89 β
I-GHDD	0.0020	0.0427	0.0817	6.9109	4.4530	18.54 β
VGED	0.0062	0.1307	0.3285	14.1893	4.8371	6.52 β
SI-URED	0.0065	0.1330	0.4857	13.6975	4.9750	2.78 β
E-STDAC	0.0060	0.1307	0.2075	11.2013	4.5520	2.56 β
I-FDFF	0.0050	0.0952	0.1613	17.8858	4.7100	4.47 β
I-AO-FDFF	0.0025	0.0471	0.1038	9.1411	4.5006	13.10 β
Kalman	0.0034	0.0753	0.0772	6.6445	4.4189	5.05 β
HGD	0.0068	0.1488	0.4429	9.3058	4.8959	2.71 β
Noise type: bell-shaped (Fig. 37 (E) and (F)), h=50ms $\beta = 0.1844$ms. Performance: red<black<blue						

The responses of the differentiators as well as the performances are shown in Table 9, Figs. 38 and 39. From Table 9, it can be seen that I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF methods still present the best responses. The reason is that unlike STD, these differentiators calculate the higher-order differentiations and then give the first-order differentiation by integrating the higher-order differentiations. Hence, the effect of the noise and numerical chattering corresponding the discontinuous part are attenuated significantly.

Comparing I-AO-STD with the E-AO-STD, it can be seen that the implicit one still presented smaller $\bar{L}_2$, $\tilde{L}_2$, L_∞, VAR and THD. It also should be noted that, in this case, the HD has almost the same THD as the ALIEN, while all other algorithms have very similar THD (excepted for Euler).

5.2.4 Performance of the differentiators in the presence of quantization

In this section, it is assumed that the signal $f(t)$ is measured with a limited resolution $q = 0.1$ (see Appendix F for basic definitions). The corresponding parameters obtained from the optimization procedure are shown in Table 10. Similarly, as in the foregoing cases, it can be seen that I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF achieve the smallest objective function ($J = 10000\bar{L}_2$), which leads to the best tracking performance. The Euler method is by far the worst of all. AO-STD, HDD and GHDD, even in their explicit form, keep quite acceptable level of performance compared to the rest of the algorithms.

Table 10: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2$
Euler (3)	No parameter	627.4098
LF (4)	c=5.7277	133.0417
E-STD (21)	L=0.7290	106.1685
I-STD (Fig. 3)	L=0.7448	105.3246
SI-STD (123)	L=0.7075	114.5138
E-URED (26)	L=0.0808, μ=16.8219	101.9228
I-URED (Fig. 5)	L=0.0885, μ=19.0403	99.4631
E-QD (25)	F=3.2796, α= 0.5979	98.8846
I-QD (60)	F=5.2542, α=0.2876	98.8066
ALIEN (14)	T=0.7617 , κ=1 , μ=5	47.6628
HD (40)*	r=1.8187	95.7837
E-AO-STD (22)**	L=3.8354	82.9877
I-AO-STD (Fig. 7) **	L=3.4485	45.6044
SI-AO-STD (122)**	L=2.8242	64.2619
E-HDD (24)**	L=3.8926	71.2434
E-GHDD (27)**	L=3.8716	70.5513
I-HDD (Fig. 9)**	L=3.2561	46.0782
I-GHDD (Fig. 10)**	L=3.2697	45.9728
VGED (35)	μ=4.9090, τ=0.4611, ω_c=1.9984, q=0.2845	95.1152
SI-URED (139)	L=0.0928, μ=97.6996	92.7848
E-STDAC (10)	α=0.9998, ϵ=0.0333	107.6598
I-FDFF (Fig. 12)	ω_s=14.6269, ω_f=1.9348, ρ=26.6241, γ=0.1479	118.4144
I-AO-FDFF (Fig. 12)	F=3.2542, ϵ=18.0793, ω_s=2.5321	52.0791
	ω_f=31.4627, α_1=11.6123, ρ=83.4065	
Kalman (Section 3.5)	$R = 7.8546 \times 10^{-4}$	45.5151
HGD (16)	L=3.5109	89.6908
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]). ** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation (output is z_1) (see (7)).		
Input signal: $\sin(t)$, Output: first-order differentiation, Performance: red<black<blue Noise-free case, under quantization 0.1, h = 50ms		

To investigate this issue precisely, the responses of the differentiators under the quantization and the

corresponding errors are shown in Figs. 40 and 41. These figures are also supported by Table 11. It can be seen that, as before, I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF presented the smallest $\bar{L}_2$, $\tilde{L}_2$, L_∞ and VAR. Moreover, these methods also show relatively small THD. Similarly, ALIEN still shows the best THD. According to Table 11, calculation time is 20 times bigger for I-AO-STD than for its explicit counterpart E-AO-STD. However, it should be noted that for the overall $\frac{10}{50\times10^{-3}} = 200$ steps the I-AO-STD only requires $45.12 \times 1.3816 = 62.34$ms. Thus, I-AO-STD needs only $\frac{62.34}{200} = 312\mu$s at each time step. Since 312μs is much smaller than the sampling time 50ms, there should not be any problem for the implementations. *However, in case of utilizing a low-cost microprocessor for the implementation, SI-AO-STD can be used to provide a compromise between the performance and the calculation time. This is a general conclusion which is also the case for other scenarios.* It can also be remarked the Euler behaves very poorly for quantization, see Fig. 41 (C), even worse than for other types of noises, compare with Figs. 18, 25 and 38. Comparing HGD and the LF in Table 11, as before, the HGD presented slightly better responses than the those of the LF.

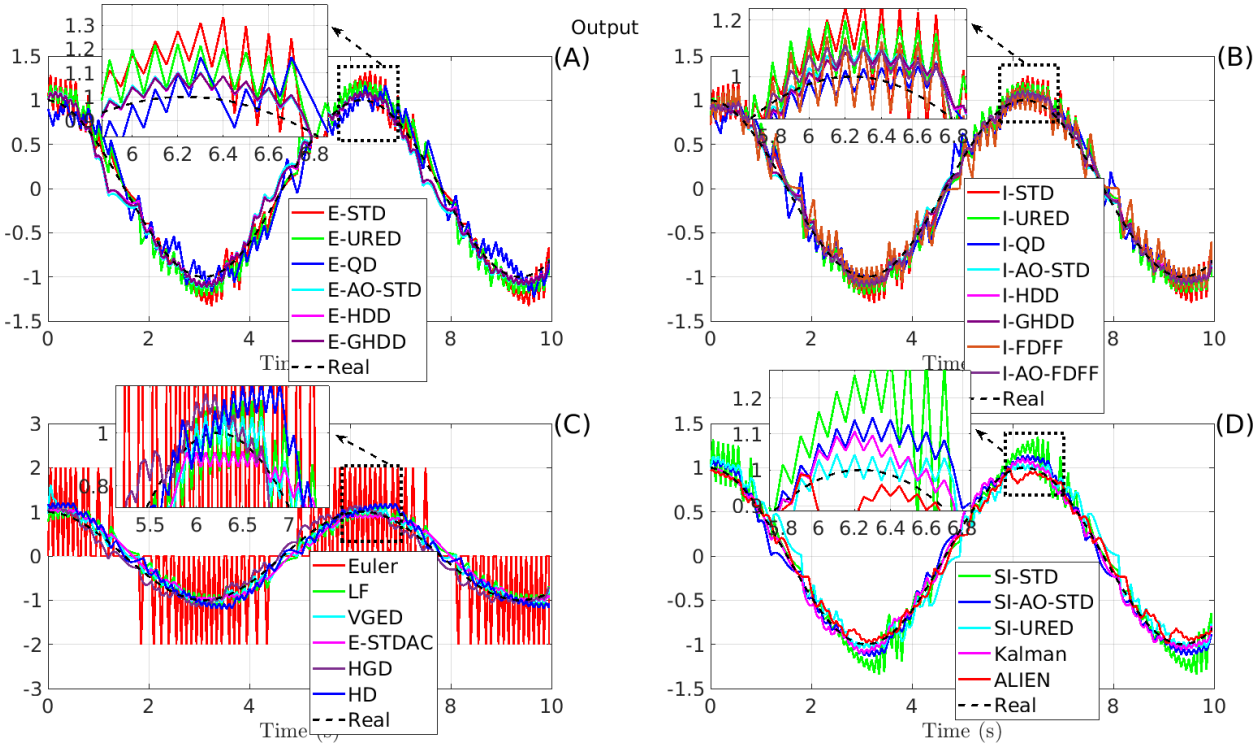

Figure 40: First-order differentiation under quantization with resolution 0.1 ($h = 50$ms).

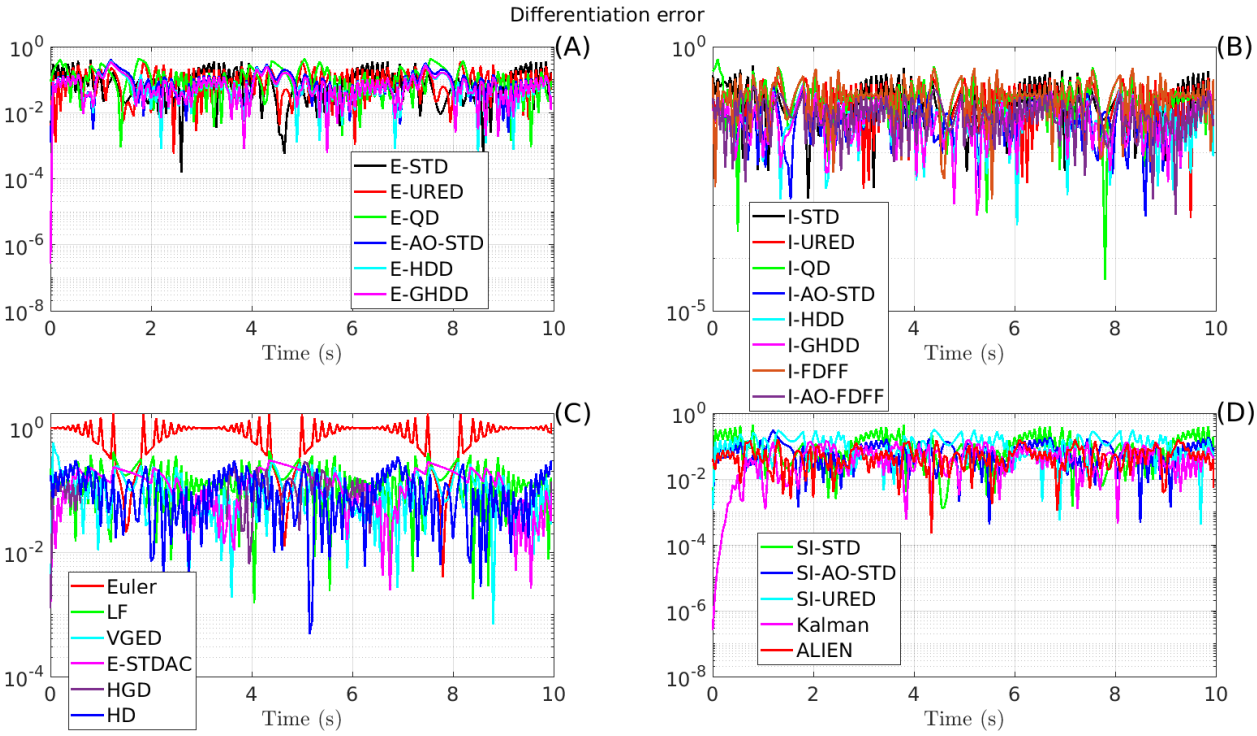

Figure 41: Error of the first-order differentiation under quantization with resolution 0.1 ($h = 50$ms).

Table 11: Results under quantization

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0627	0.9258	1.7325	262.0000	13.6779	1.00 β
LF	0.0133	0.2599	0.4269	39.6138	5.1809	1.07 β
E-STD	0.0106	0.1978	0.3973	31.7769	5.0247	1.18 β
I-STD	0.0105	0.1871	0.4406	34.4403	4.8951	1.52 β
SI-STD	0.0115	0.2099	0.4559	33.7188	5.1390	1.13 β
E-URED	0.0102	0.1959	0.3193	32.5109	4.9209	1.44 β
I-URED	0.0099	0.1881	0.3301	32.7034	4.8130	45.61 β
E-QD	0.0099	0.2066	0.3370	24.4212	4.7722	1.43 β
I-QD	0.0099	0.2082	0.4607	20.3679	4.7310	2.88 β
ALIEN	0.0048	0.1017	0.1595	9.4103	3.5737	22.47 β
HD	0.0096	0.1996	0.3490	20.7500	4.0643	8.31 β
E-AO-STD	0.0083	0.1806	0.3987	12.8129	4.7367	2.86 β
I-AO-STD	0.0046	0.0953	0.1853	10.9223	4.5183	45.12 β
SI-AO-STD	0.0064	0.1372	0.3046	12.6287	4.6805	3.20 β
E-HDD	0.0071	0.1534	0.3658	12.8369	4.7297	3.35 β
E-GHDD	0.0071	0.1525	0.3516	12.2091	4.7135	3.87 β
I-HDD	0.0046	0.0969	0.2001	10.6946	4.4637	44.07 β
I-GHDD	0.0046	0.0970	0.1855	10.1942	4.5098	41.94 β
VGED	0.0095	0.1972	0.6217	22.1858	5.0433	18.72 β
SI-URED	0.0093	0.1984	0.2988	15.4889	4.6394	1.43 β
E-STDAC	0.0108	0.2383	0.2861	11.5551	4.6647	1.77 β
I-FDFF	0.0118	0.2228	0.4024	40.1810	5.2965	1.96 β
I-AO-FDFF	0.0052	0.1060	0.1785	14.5573	4.3919	27.96 β
Kalman	0.0046	0.0962	0.1497	10.4910	4.4728	5.66 β
HGD	0.0090	0.1925	0.3819	15.8688	4.6620	2.07 β
Under quantization 0.1, h=50ms $\beta = 1.3816$ms. Performance: red<black<blue						

5.3 Effect of the sampling period on the differentiators

The previous simulations were conducted for a relatively large sampling period ($h = 50$ms) compared with the signal frequency. The purpose of this section is to investigate the responses for other sampling periods $h \in [0.1, 100]$ms. In this context, the objective functions are shown in Figs. 42 to 46 for different sampling periods. In this case, the gains are tuned as reported in Table 12, for $h = 10$ms and SNR=∞. One infers

from these figures that in a noise-free case, increasing the sampling period deteriorates the performances of all differentiators, except for the criteria VAR and L_∞. Note that in this noise-free case, the response of the Euler differentiation can be considered as the optimal one.

Table 12: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2$
Euler (3)	No parameter	1.0926
LF (4)	c=999991.1108	1.0928
E-STD (21)	L=0.8383	4.1557
I-STD (Fig. 3)	L=137.0017	1.0926
SI-STD (123)	L=0.8338	5.0663
E-URED (26)	L=0.0410, μ=410.1379	4.1527
I-URED (Fig. 5)	L=181.0637, μ=202.4230	1.0926
E-QD (25)	F=5.1201, α= 3.5689	35.2396
I-QD (60)	F=1999.7762, α=41.8930	1.1646
ALIEN (14)	T=0.0900 , κ=4 , μ=3	2.9676
HD (40)*	r=1.2200	2.1760
E-AO-STD (22)**	L=1.4859	4.1535
I-AO-STD (Fig. 7) **	L=0.6714	0.9352
SI-AO-STD (122)**	L=1.0277	2.3751
E-HDD (24)**	L=1.3011	3.2352
E-GHDD (27)**	L=1.3438	3.2462
I-HDD (Fig. 9)**	L=0.6346	1.9325
I-GHDD (Fig. 10)**	L=0.6468	1.9214
VGED (35)	μ=1.4324, τ=138.3938, ω_c=17.2907, q=2.8214	2.0899
SI-URED (139)	L=34.6716, μ=0.9485	3.5217
E-STDAC (10)	α=0.8270, ϵ=0.0100	3.3841
I-FDFF (Fig. 12)	ω_s=999.9819, ω_f=564.3257, ρ=659.3234, γ=194.0534	1.3111
I-AO-FDFF (Fig. 12)	F=198.8190, ϵ=887.4548, ω_s=5.4169	0.5396
	ω_f=1684.9944, α_1=4359.4855, ρ=4451.7299	
Kalman (Section 3.5)	$R = 10^{-6}$	0.5333
HGD (16)	L=8.8329	5.9314
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]). ** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation (output is z_1) (see (7)).		
Input signal: $\sin(t)$, Output: first-order differentiation, Performance: red<black<blue Noise-free case, $h = 10$ms		

From Fig. 45 it can be seen that the variations are remain constant except for the I-QD and E-URED.

Keeping in mind that the implicit methods does not present the numerical chattering inherently.

Comparing Fig. 42 (A) and Fig. 42 (B), we recover here the fact that the implicit method allows the designer to choose larger sampling times, a fact already noticed in the context of sliding-mode control [53, 54, 60]. For example, Fig. 42 (A) shows that for $h = 10^{-2}$s, $\bar{L}_2$ for the explicit methods is always higher than 3×10^{-4} which is equal to the performance of the implicit methods in $h = 3 \times 10^{-2}$s. It can be concluded that for this specific performance, implicit methods allow the designer to select three times larger sampling period. *Therefore, it is inferred that implicit methods can reduce the calculation burden by increasing the sampling time.* A similar tendency can also be observed for the VAR in Fig. 45. Furthermore, *we infer that I-AO-STD and I-URED have the most robust behavior with respect to variation in h.*

Fig. 45 (D) shows that the ALIEN does not show the best variation for all sampling times anymore. The reason is that the ALIEN is tuned for $h = 10$ms and by changing the sampling time, it needs re-tuning. Also note that all methods possess a similar harmonic attenuation, see THD in Fig. 46 when h varies.

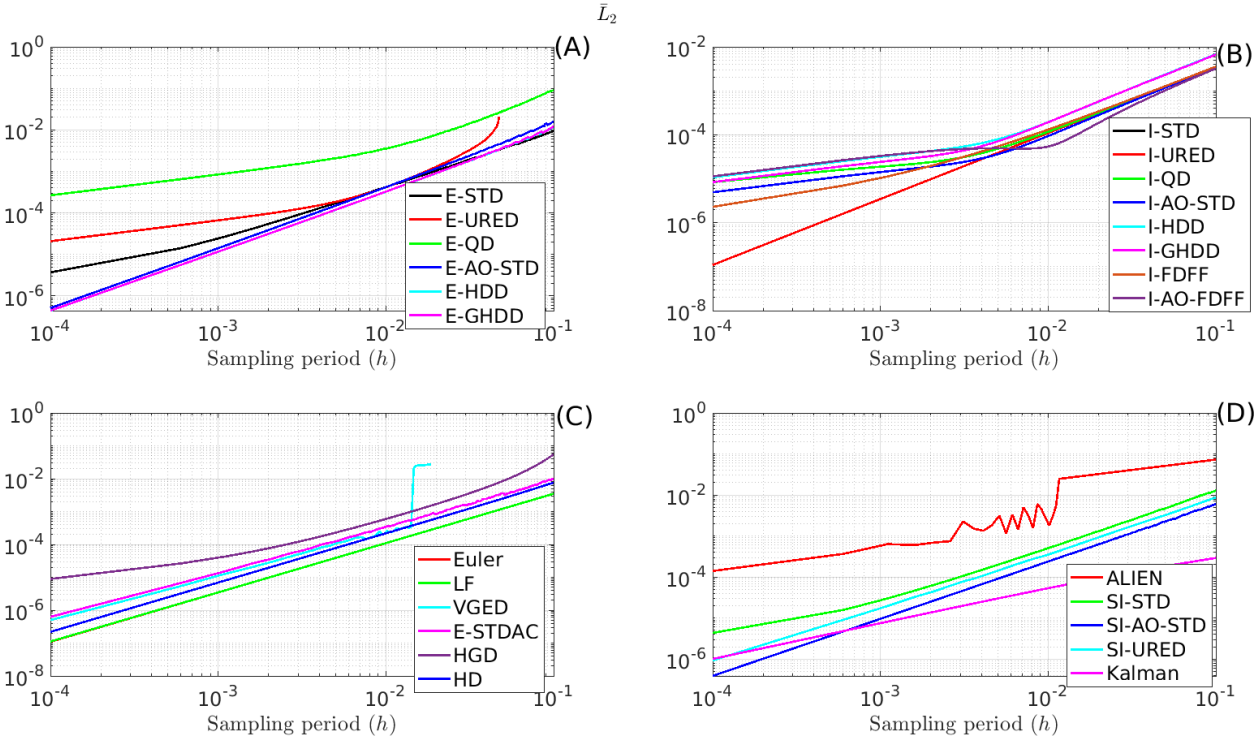

Figure 42: $\bar{L}_2$ for the first-order differentiation with different sampling periods (SNR=∞).

From Fig. 44 (A) and (B) it can be seen that the implicit schemes are more accurate than the explicit counterparts.

As it can be seen, for $h > 20$ms the output of the VGED is always zero. The reason is that this differentiator is tuned for $h = 10$ms, and for larger sampling times, this differentiator calculates complex outputs. In other words, for large sampling times, a negative value may be obtained for γ which leads to a complex α. Hence, the output will be complex and invalid.

Figs. 42 to 44 show that global tendency is that all algorithms performances decrease as h increases. Except for the L_∞ of the I-QD, I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF where the L_∞ decreases for $h < 2$ms. This shows that those differentiators should be tuned for smaller sampling time.

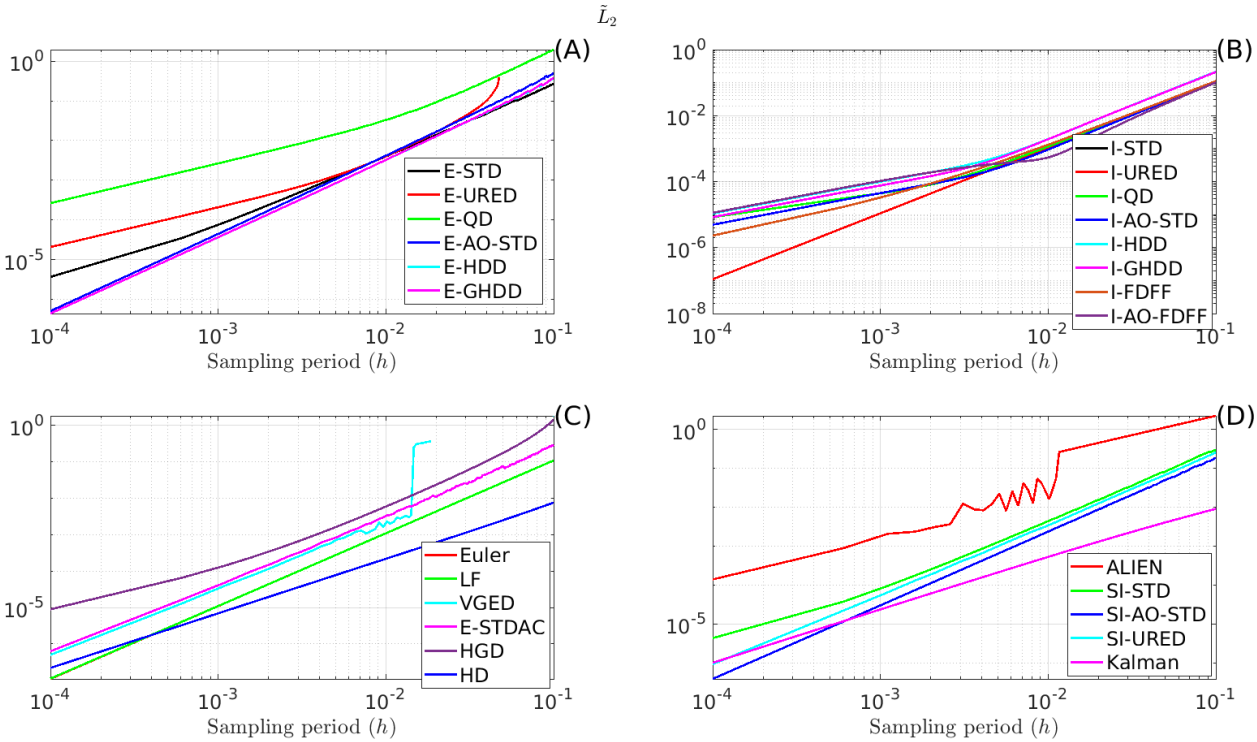

Figure 43: $\tilde{L}_2$ for the first-order differentiation with different sampling periods (SNR=∞).

All Figs. 42 to 44 (B) demonstrate the existence of two main linear behaviors for $h < 10^{-2}$s and $h > 10^{-3}$. Comparison of HGD and LF in Figs. 42 to 44 demonstrates the superiority of the LF for this special case. Hence, LF presents a more robust behavior w.r.t changing the sampling time when considering the $\bar{L}_2, \tilde{L}_2, L_\infty$.

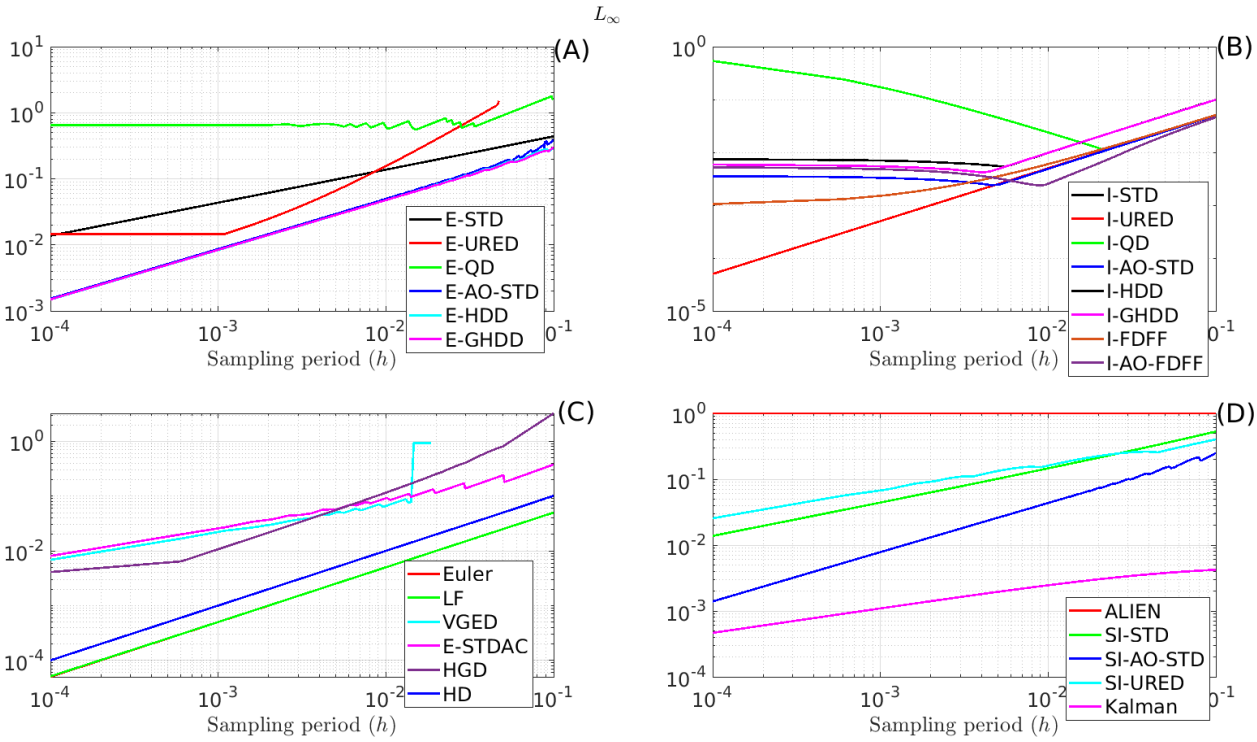

Figure 44: L_∞ for the first-order differentiation with different sampling periods (SNR=∞).

Fig. 45 shows that variation for the VGED is zero for large h. In fact, the VGED generates complex numbers for $h > 60$ms. We infer from Fig. 45 that all differentiators behave in the same way w.r.t the h, when variation is considered except for the I-QD where the var slightly decreases with increasing h. It should also be noted that E-URED is unstable for $h > 45$ms. In fact, the tuning algorithm selected a lage μ for this differentiator to improve its performance for $h = 10$ms. However, with this large gain, this differentiator is unstable for larger sampling times.

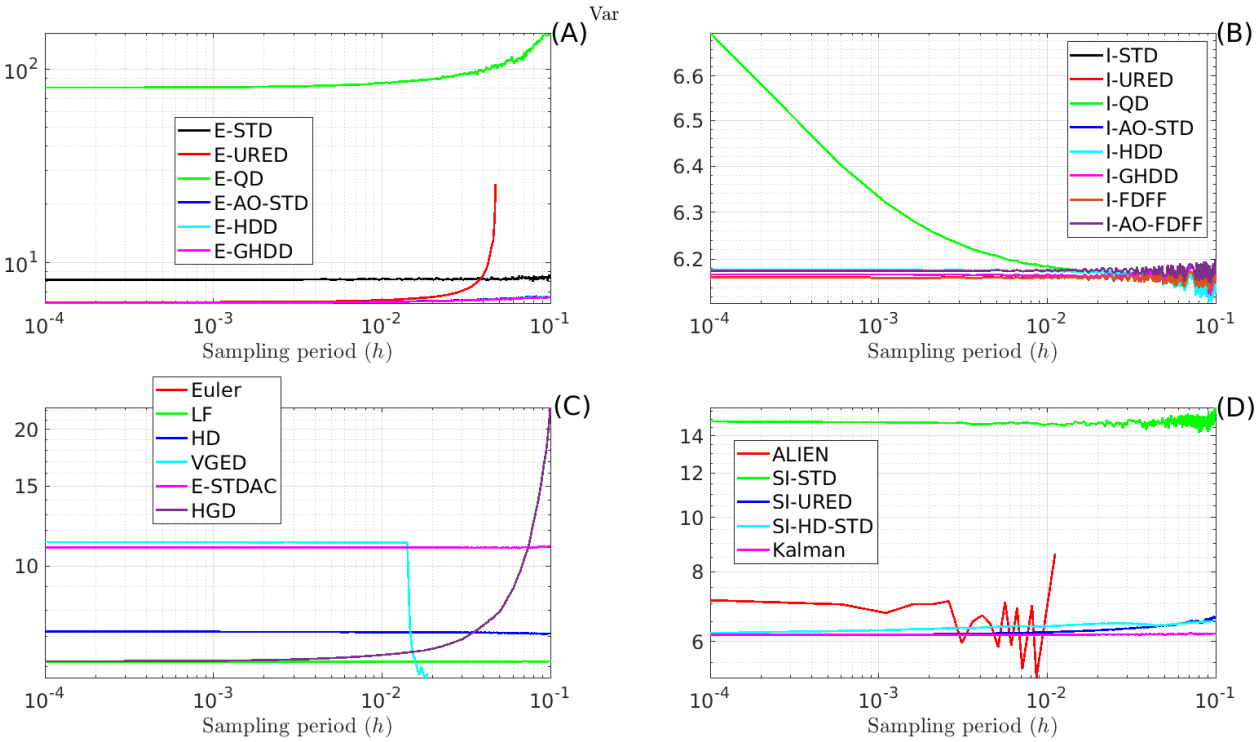

Figure 45: Variation for the first-order differentiation with different sampling periods (SNR=∞).

We infer from Fig. 46 that all schemes evolve in a similar way when h increases. Thus the influence of the sampling period seems to be independent of the kind of differentiate (explicit vs implicit, sliding-mode vs linear or ALIEN).

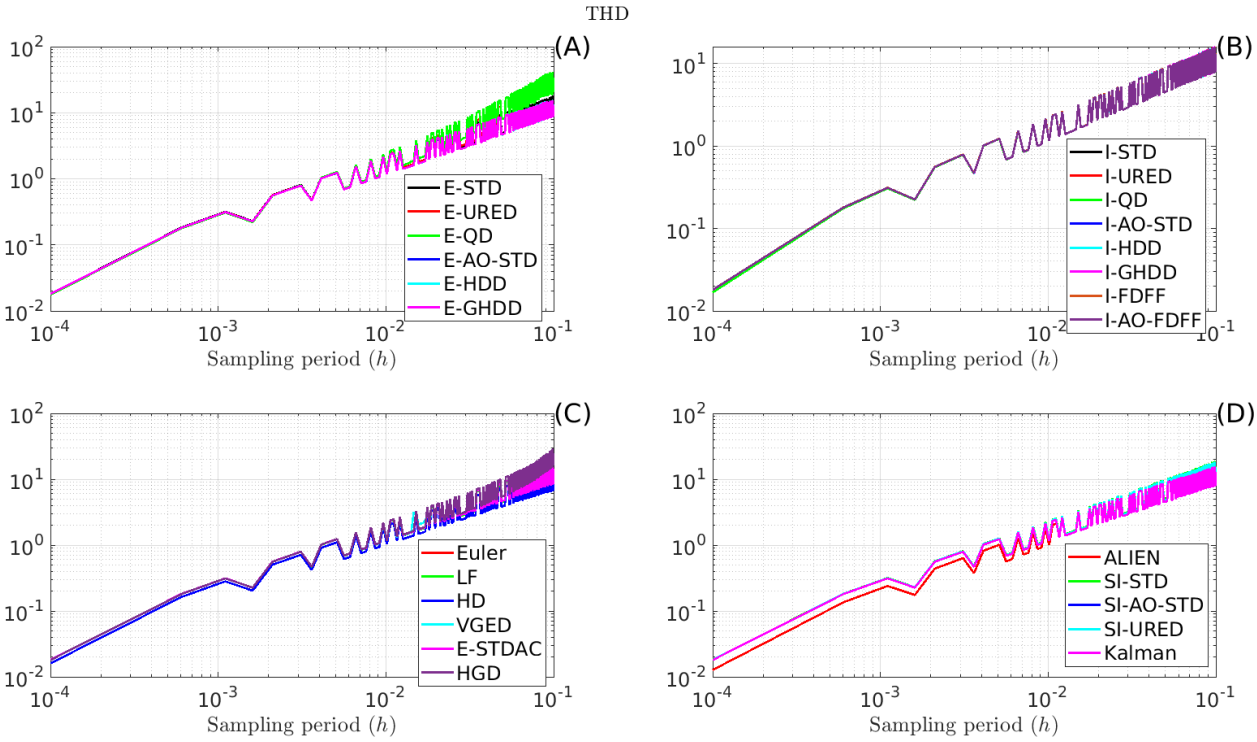

Figure 46: THD for the first-order differentiation with different sampling periods (SNR=∞).

5.4 Effect of the frequency of the input signal on the differentiators

We have investigated the effect of the input signal $f(t)$ frequency on the differentiators. However, it seems that the parameters are extremely sensitive to the input frequency. The reason is that the optimization procedure tunes the parameters, such that only specific low-frequency signals are allowed to pass through the differentiator, and higher frequencies will be blocked. As a result, with increasing the frequency of the input signal, the differentiator considers that as a high-frequency noise. Therefore, with re-tuning the parameters for an specific signal, the aforementioned results will still be valid. Thus, it has been decided not to repeat this procedure for different input frequencies. This issue should be addressed in closed-loop studies, however.

5.5 Behavior of the differentiators in the presence of initial error

As it was mentioned before, it is not possible to set the initial conditions for some differentiators, e.g., ALIEN differentiators. On the hand, the number of initial conditions for differentiators are not necessarily equal, due to the fact that their dimensions as dynamical systems vary from one differentiator to another. For example, the STD has two initial conditions $(f(0), \dot{f}(0))$ while an eight-order AO-STD has nine initial conditions $(f^{(i)}(0), i = 0, 1, \ldots, 8)$. Hence, considering the first-order differentiation, an AO-STD may present better transient responses than those of the STD where higher-order differentiations are not zero at the initial

time[1].

To provide a fair comparison, it is assumed that all initial conditions are set to zero ($f^{(i)}(0) = 0, i = 0, 1, \dots$), which means that all methods show the initial error for the selected input $\sin(t)$. The parameters obtained form the tuning procedure are given in Table 13. According to Table 13, the VGED presents the best response. The internal variables of this differentiator are shown in Fig. 48. It can be seen that the exponent $\gamma(t)$ of this differentiator is not constant. Compared to other differentiators, this differentiator has one of the highest number of parameters. It seems that the number of parameters along with the adaptation mechanism improve the responses of the VGED for the specific condition. One can see in Fig. 48 that the amplitude of $|f_f|$ is larger at the beginning of the simulation. The reason is that because of non-zero initial error, the input acts as a disturbance. This disturbance increases $|f_f|$. The adaptation mechanism uses $|f_f|$ to tune the exponent of the VGED $\alpha(t)$ and improves the transient responses. However, as it will be shown in Figs. 49 to 52, with the selected parameters, the VGED does not necessarily present the best responses for other conditions.

From Table 13, it can be observed that implicit methods still achieved one of the best performances when compared to other methods. The responses of the differentiators are shown in Fig. 47. Comparing this figure with its zero-initial-error counterpart Fig. 18, it can be seen that all differentiators present initial error at the initial time. As it was expected, the Euler method still presents the worst responses in the presence of noise.

[1]Note that both STD and AO-STD can calculate the first-order differentiation.

Table 13: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2(e_k)$
Euler (3)	No parameter	400.7426
LF (4)	c=7.7652	125.2666
E-STD (21)	L=0.7621	119.3866
I-STD (Fig. 3)	L=0.7375	114.6544
SI-STD (123)	L=0.7105	120.0440
E-URED (26)	L=0.1526, μ=17.2523	111.7988
I-URED (Fig. 5)	L=0.1842, μ=19.2301	106.3642
E-QD (25)	F=3.6398, α= 0.5345	114.1485
I-QD (60)	F=4.4258, α=108.0366	112.2623
ALIEN (14)	T=0.5020 , κ=1 , μ=2	137.2458
HD (40)*	r=2.5653	150.2620
E-AO-STD (22)**	L=20.4049	160.8194
I-AO-STD (Fig. 7) **	L=14.3801	102.5989
SI-AO-STD (122)**	L=9.7812	120.3473
E-HDD (24)**	L=15.9819	137.7374
E-GHDD (27)**	L=20.4050	144.8012
I-HDD (Fig. 9)**	L=14.3671	96.9191
I-GHDD (Fig. 10)**	L=15.7542	98.9933
VGED (35)	μ=6.3813, τ=5.9048, ω_c=12.1897, q=0.0011	82.4729
SI-URED (139)	L=0.1779, μ=96.4825	104.6239
E-STDAC (10)	α=0.7896, ϵ=0.0018	160.3342
I-FDFF (Fig. 12)	ω_s=59.4160, ω_f=22.6361, ρ=12.6443, γ=0.0001	115.2277
I-AO-FDFF (Fig. 12)	F=53.7596, ϵ=22.9829, ω_s=3.7986	93.5342
	ω_f=37.5615, α_1=54.2768, ρ=29.4002	
Kalman (Section 3.5)	$R = 3.5920 \times 10^{-8}$	366.6154
HGD (16)	L=4.0992	159.2564
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]).		
** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation		
Input signal: $\sin(t)$, Output: first-order differentiation.		
Noise type: white, SNR=30dB, h = 50ms. Performance: red<black<blue		
All initial conditions of all the differentiators are set to zero. It results in initial error.		

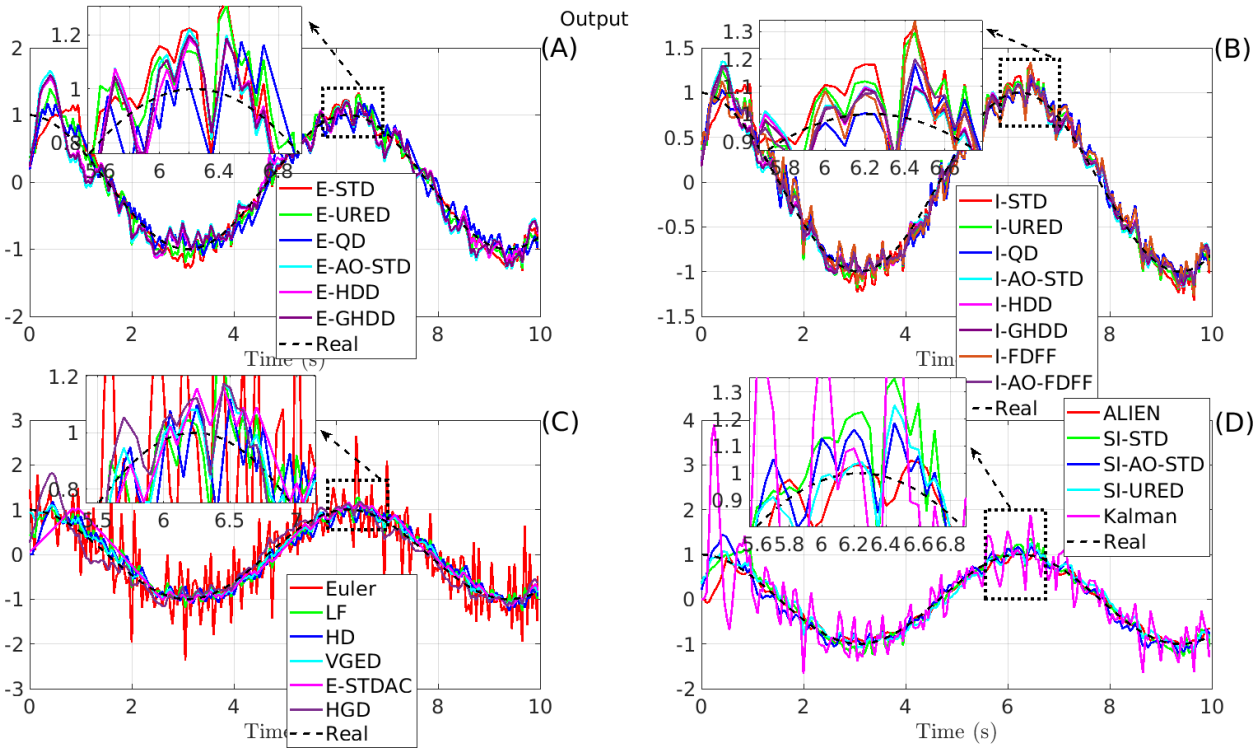

Figure 47: Output of the differentiators in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms, SNR=30dB).

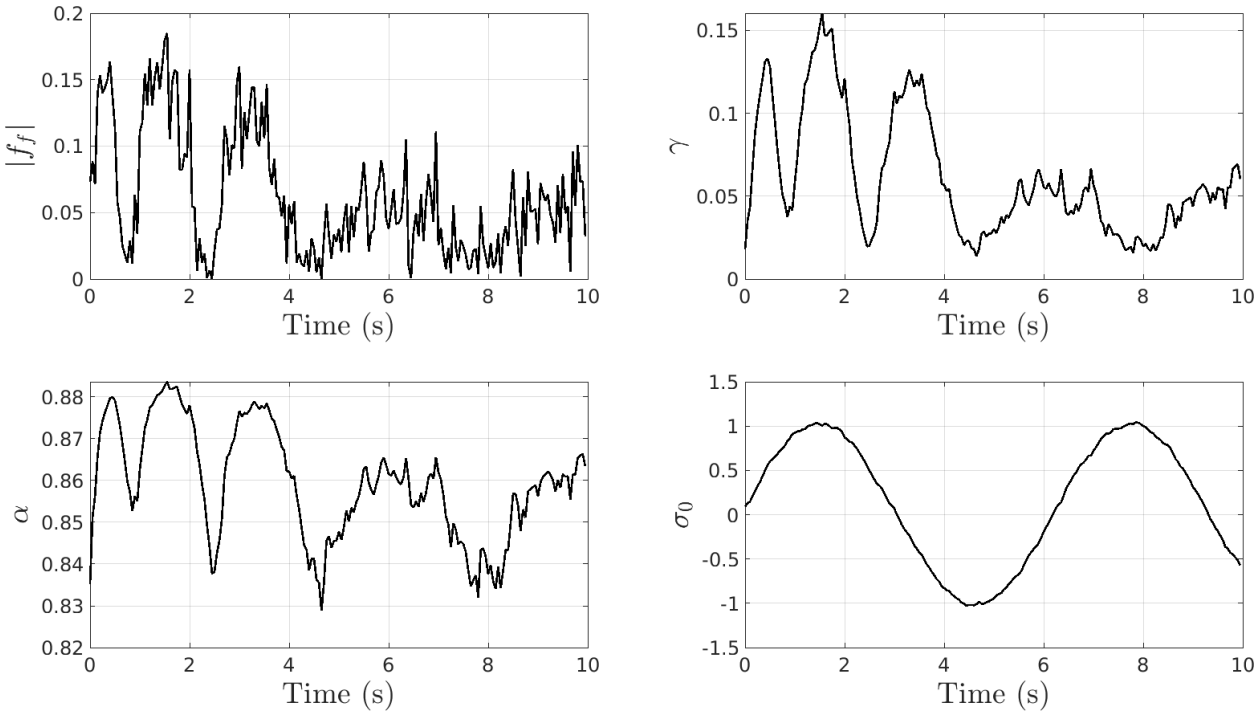

Figure 48: Internal variables of the VGED. Parameters are shown in Table 13 ($h = 50$ms, SNR=30dB).

The criteria in the presence of the initial error are provided in Figs. 49 to 53. These figures show that for large SNRs (here SNR>50dB) as expected, the Euler method shows the best performances except for the variation and THD in Figs. 52 and 53, respectively.

From Figs. 49 and 50 one can see that the Euler method presents the worst performance for small SNRs. Moreover, the implicit methods, in general, possess the best performances. Among them, I-HDD, I-GHDD, and I-AO-FDFF seem to have the best performances for the whole SNRs. Moreover, while the I-AO-STD shows the same performance as the I-HDD, I-GHDD, and I-AO-FDFF, its $\bar{L}_2$ and $\tilde{L}_2$ performances suddenly diverge and get worse for SNR>60, becoming equal to that of I-QD (Figs. 49 and 50 (B)).

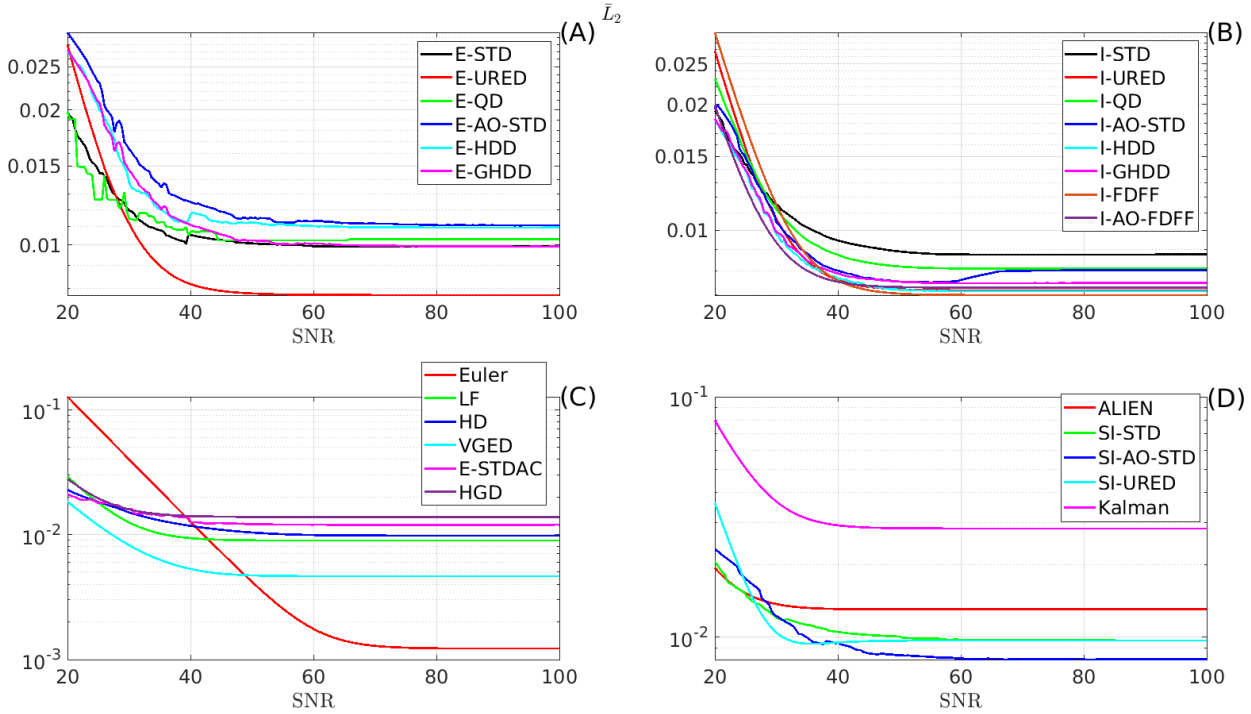

Figure 49: $\bar{L}_2$ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms).

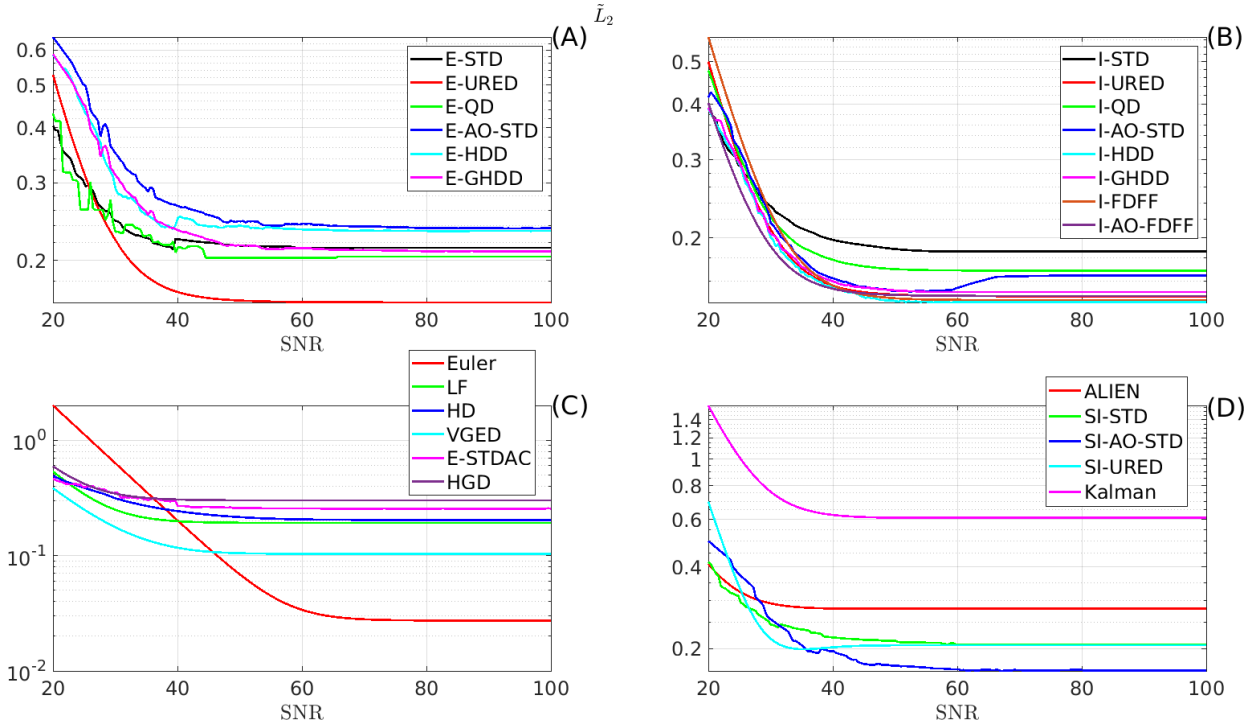

Figure 50: $\tilde{L}_2$ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms).

The L_∞ criterion is shown in Fig. 51. As expected, the Euler method presented the smallest L_∞ for large SNRs (SNR>60dB). In fact, unlike others, the bandwidth of the Euler method is infinite. This improves its transient response significantly. However, this infinite bandwidth deteriorates the performances for small SNR (SNR<60dB) since the noisy components can also pass through the differentiator.

It is also interesting to see that some differentiators in Fig. 51 (A) and (B) show the minimum L_∞ for SNR≈28dB. This kind of observation is not strange in cases where a stochastic noise is considered. Note that since an initial seed is considered for the white noise generator, all of them show similar behaviors. Apart from the Euler, the SI-STD and I-STD show generally the best L_∞. Considering the explicit methods, the E-STD and E-URED also show small L_∞. Hence, based on this observation from Fig. 51, it can be concluded that *smaller number of initial conditions may result in better response to input disturbances since a differentiator with higher number of state variables (initial conditions) needs more time to respond to the disturbances. However, as it was seen before, a higher-order differentiator can provide better responses in the absence of input disturbances, which leads to better steady-state performances.*

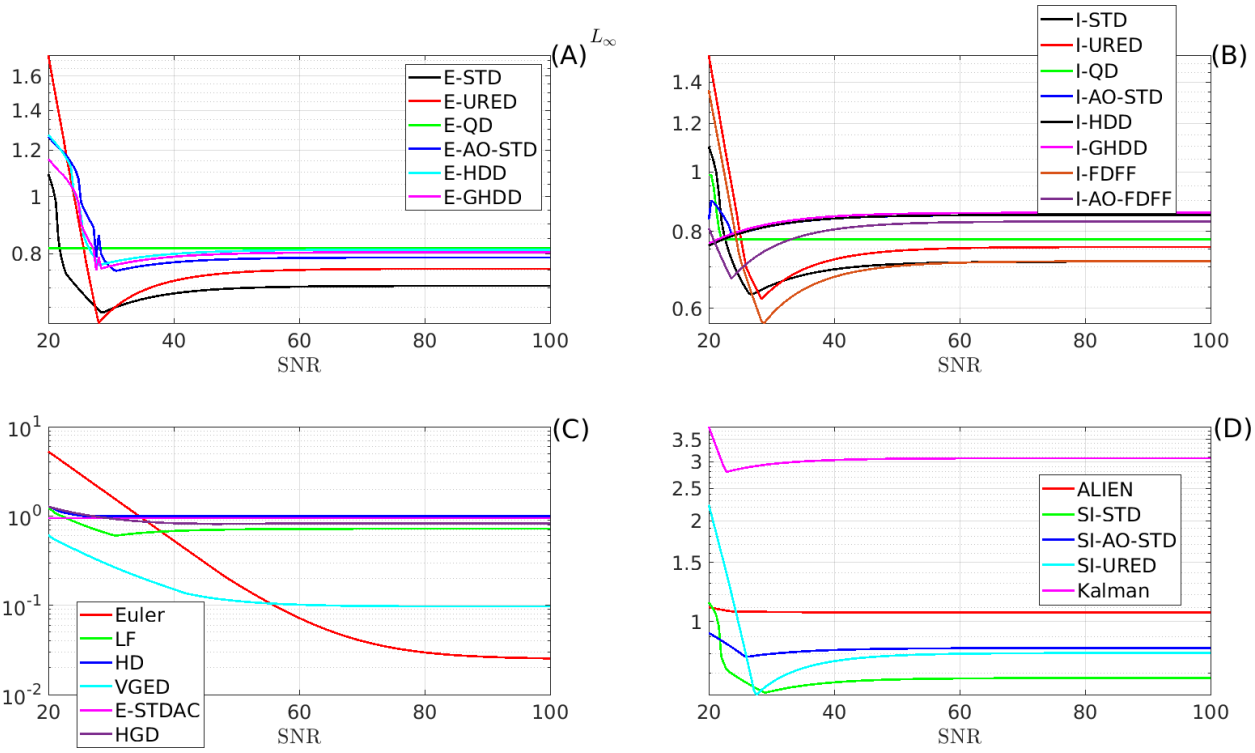

Figure 51: L_∞ for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms).

Fig. 52 shows that in general, implicit methods present the smallest variations. Other methods such as SI-URED and ALIEN differentiator also shows small variations where the ALIEN presents the smallest variation for all SNRs.

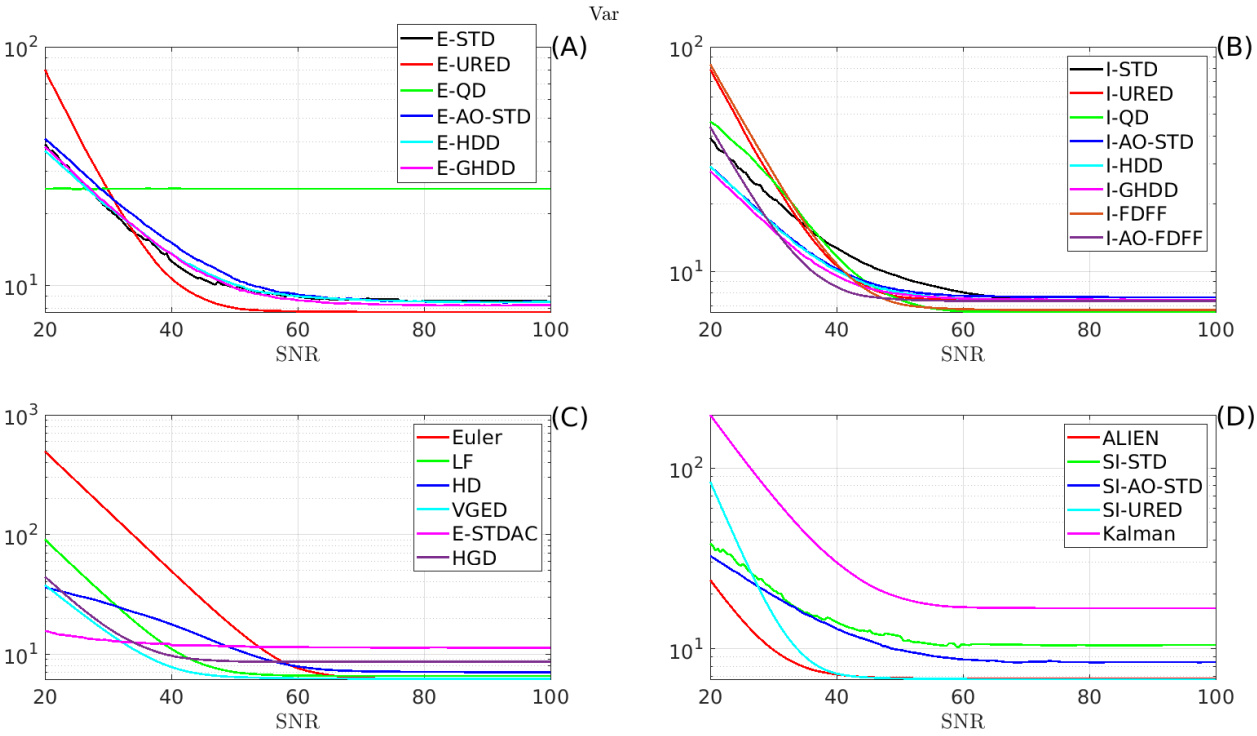

Figure 52: Variation for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms).

THDs are also shown in Fig. 53. The first observation is that, as before, the ALIEN shows the best THD. Comparing Fig. 53 (A) and (B), it can be seen that the implicit methods in general show better THDs compared to that of the explicit schemes. Also, semi-implicit schemes, in general, remain between the explicit and implicit methods. Comparing the LF and HGD, one sees that the LF presents better behavior in handling the initial error.

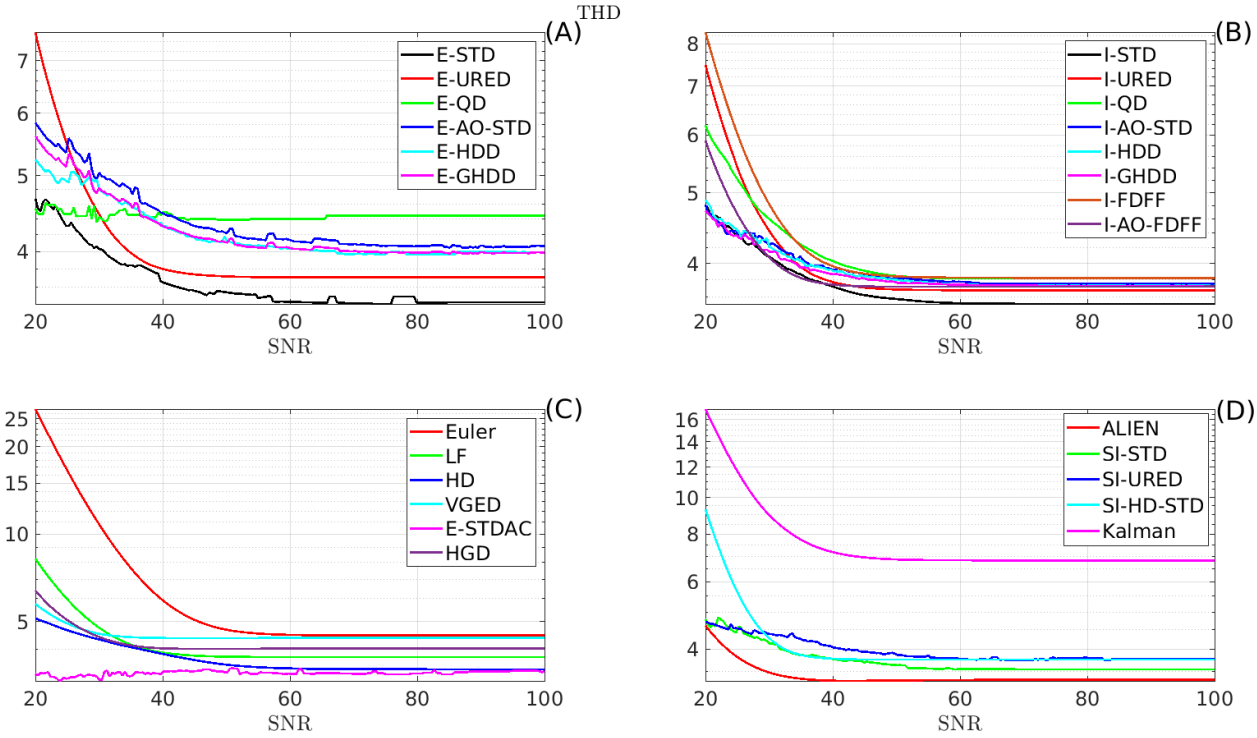

Figure 53: THD for the first-order differentiation with respect to SNR of the white noise in the presence of initial error. Parameters are shown in Table 13 ($h = 50$ms).

5.6 Higher-order differentiation

As mentioned before, some applications require higher-order differentiation, for instance, for calculating the sliding variable in a first-order SMC [54, 57]. In order to calculate higher-order differentiation, the outputs of the arbitrary-order differentiators have been compared with the cascade configuration of the first-order differentiators in the presence of white noise. The block diagram of the cascade configuration of differentiators is shown in Fig. 54. The cascade configuration calculates the higher-order differentiations in an online manner. In other words, at each time-step, the output of each stage is considered as the input of the next stage. Note that among the differentiators, only AO-STD, I-HDD, I-GHDD, I-AO-FDFF, HD and ALIEN can calculate the higher-order differentiation. Consequently, higher-order differentiation for other differentiators e,g., STD, URED, QD, LF and Euler, are calculated using the cascade configuration throughout this work.

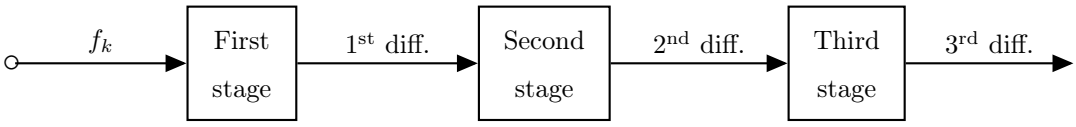

Figure 54: Flowchart of the third-order cascade configuration of differentiators

To provide a fair comparison, the parameters of the differentiators are tuned for the third-order differentiation ($h = 10$ms and SNR=90dB) and given in Table 14. The output of the differentiators for calculating the third-order differentiation of the $\sin(t)$ under the aforementioned condition is provided in Fig. 55. Comparing all the outputs in Fig. 55, it can be seen that I-AO-STD, I-HDD, I-GHDD, I-URED, and I-AO-FDFF provided the most accurate responses. It can be seen that most of the differentiators show irregular responses. The reason is that with increasing the differentiation-order, the effect of noise will be amplified. Thus, even for this relatively large SNR=90dB, the noise deteriorated the performances significantly.

It also should be mentioned that the tuning algorithm selects small gains for the E-AO-STD, E-HDD and E-GHDD such that they do not present chattering for the specific input (here $\sin(t)$). With these gains, these methods will not be exact anymore. Hence, the E-AO-STD, E-HDD and E-GHDD presented one of the smallest variations. With these small gain, the responses of the E-AO-STD, E-HDD and E-GHDD are almost insensitive to the SNR because they will not respond to the high-frequency components (corresponding to noise) in the input with these small gains (see Fig. 59 (A)).

In fact, AO-STD, HDD, GHDD, AO-FDFF calculate higher-order differentiations and then estimate the desired differentiation by integration of the higher-order differentiation. Thus, they act as a moving average filter. Since moving average filter is a low-pass filter, an inherent low-pass filtering always exists in these methods which filters high-frequency components corresponds to both input noise and the numerical chattering. As we can see from Fig. 59, E-AO-STD always presents less chattering than E-STD. Considering this fact, it is not reasonable to use pre-filtration stages before the AO-STD. However, pre-filtration might be necessary for STD, VGED and STDAC because the output will be calculated directly without integrating the higher-order differentiations.

The Objective functions with respect to changing the SNR are shown in Figs. 56 to 60. As it is mentioned before, higher-order differentiation is more sensitive to noise than the first-order differentiation such that for SNR$<$70dB all the methods present too much fluctuations. Hence, the objective functions are shown for SNR$>$70 to provide realistic conclusions. Similar to the first-order differentiation, it can be seen that the implicit methods still present appropriate responses except for the I-QD.

According to these results, I-URED, I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF have presented the best responses. It is also interesting to note that E-HDD, E-GHDD and E-AO-STD also presented appropriate responses and the reason is that a relatively small sampling period ($h = 10$ms) is considered in this simulation. The effect of the considered noise (SNR=90db) is amplified for the third-order differentiation. Thus, the main source for the performance degradation is not numerical chattering.

Note that I-AO-STD requires an online solver to solve a polynomial equation at each time-step as explained in Section 3.3.4. Thus, for applications with limited calculation resources it might be necessary to use cascade configuration of I-STD rather than I-AO-STD. From Figs. 56 to 60 (A), it can be seen that among explicit methods, E-AO-STD, E-HDD and E-GHDD show the best responses, though less good than their implicit counterparts. From the chattering point of view, Fig. 59 (and also Fig. 56 (D)) shows that

ALIEN presents the smallest variation. Moreover, Fig. 60 (B) and (D) show that ALIEN also shows the best THD. Among the explicit methods, HD in Fig. 60 (C) shows the best THD, close to ALIEN.

Table 14: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000\bar{L}_2$
Euler (3)	No parameter	31830.4094
LF (4)	c=16.9388	53.5800
E-STD (21)	L=0.7532	64.0427
I-STD (Fig. 3)	L=0.7728	48.9858
SI-STD (123)	L=0.7281	71.4082
E-URED (26)	L=1.1782, μ=0.1932	103.6176
I-URED (Fig. 5)	L=2.4025, μ=81.7318	35.0034
E-QD (25)	F=3.8498, α= 0.4715	167.3635
I-QD (60)	F=15.2169, α=3.3914	114.4203
ALIEN (14)	T=0.8376 , κ=2 , μ=1	23.3915
HD (40)*	r=1.2621	165.7308
E-AO-STD (22)**	L=2.4167	41.6795
I-AO-STD (Fig. 7) **	L=4.0095	35.2477
SI-AO-STD (122)**	L=2.2759	44.6142
E-HDD (24)**	L=1.9111	42.5058
E-GHDD (27)**	L=1.9649	41.0811
I-HDD (Fig. 9)**	L=4.2086	36.6057
I-GHDD (Fig. 10)**	L=3.3526	35.3997
I-AO-FDFF (Fig. 12)	F=9.8148, ϵ=38.2115, ω_s=20.6008	34.0470
	ω_f=159.0598, α_1=129.0693, ρ=149.3592	
Kalman (Section 3.5)	$R = 3.8107 \times 10^{-9}$	31.0850
HGD (16)	L=7.4618	92.9094
* Three first-order differentiators utilized in the cascade configuration i.e., Fig. 54.		
** Third-order HD which is utilized to calculate the third-order differentiation (see [8]).		
*** Third-order AO-STD ($n = 3$) which is utilized to calculate the third-order differentiation (output is z_3) (see (7)).		
Input signal: $\sin(t)$, Output: third-order differentiation		
Noise type: white, SNR: 90dB, h = 10ms		

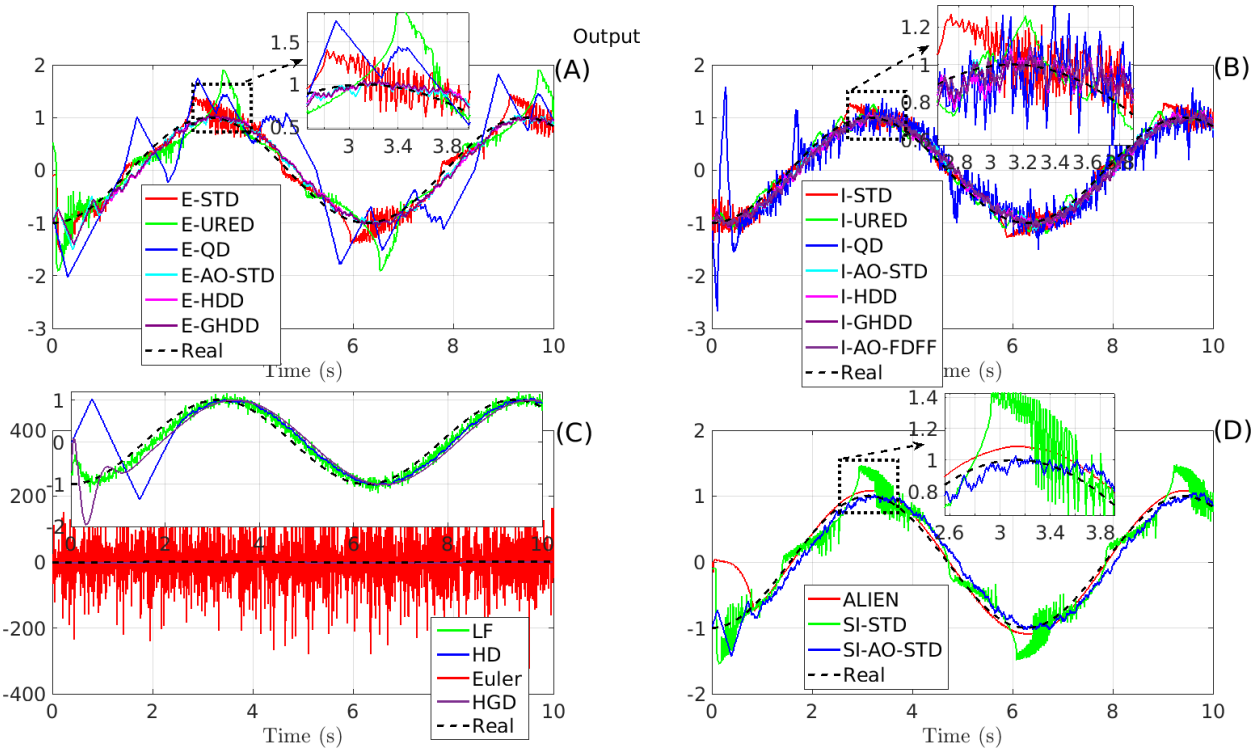

Figure 55: Third-order differentiation in the presence of white noise. Parameters are shown in Table 14 ($h = 10$ms, SNR=90dB).

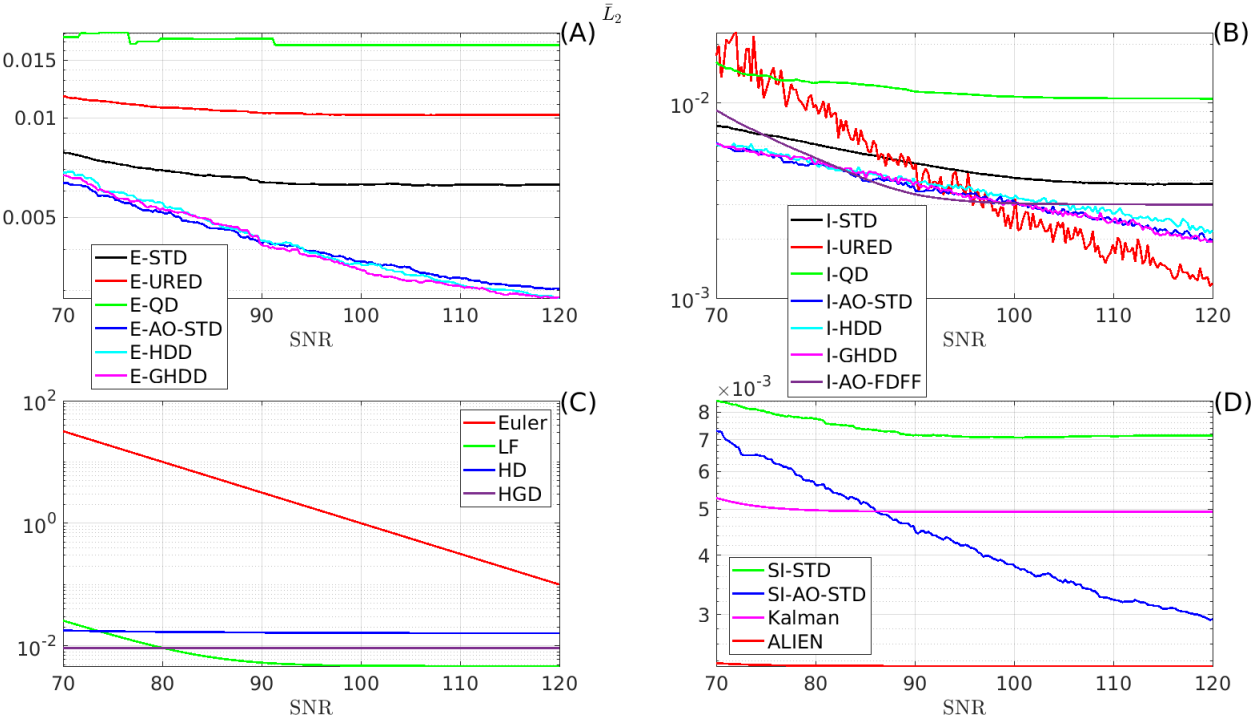

Figure 56: $\bar{L}_2$ for the third-order differentiation under different SNRs (h =10ms).

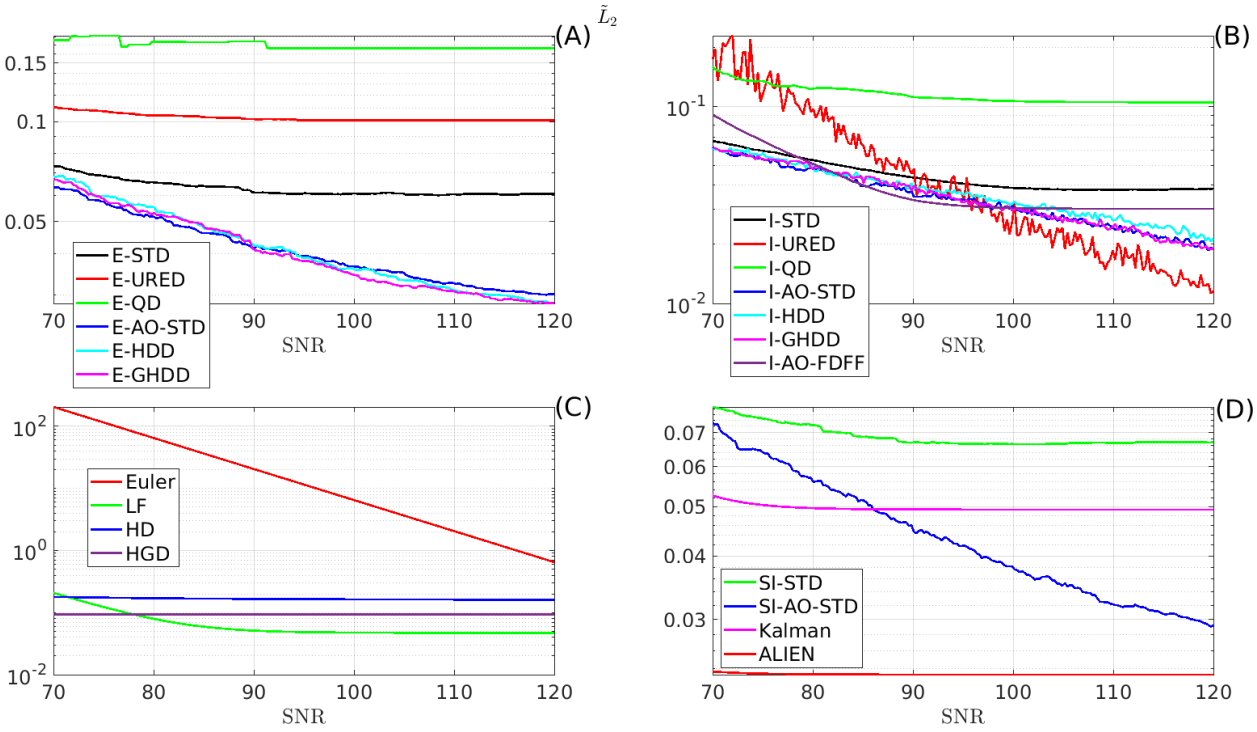

Figure 57: $\tilde{L}_2$ for the third-order differentiation under different SNRs (h =10ms).

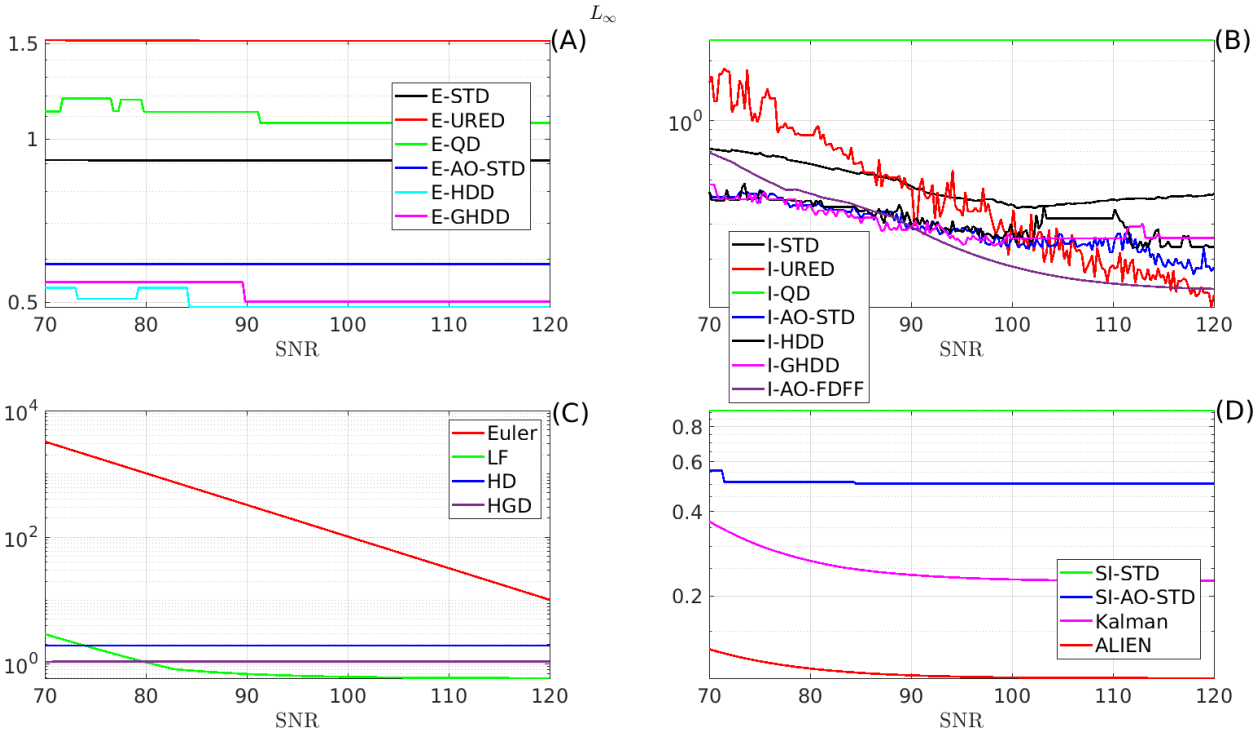

Figure 58: L_∞ for the third-order differentiation under different SNRs (h =10ms).

According to Fig. 58 (B), the I-AO-FDFF behaves quite well for large SNR.

For $\bar{L}_2$, $\tilde{L}_2$ and L_∞ criteria, the group (AO-STD, HDD and GHDD) is the best in both explicit and implicit methods. SI-AO-STD shows good performances as well (Figs. 56 to 58 (D)).

From Fig. 59 (A) and (B), for "small" SNRs, the best are I-AO-STD, I-HDD and I-GHDD. On the other hand, for the "large" SNRs, the best is I-URED. Furthermore, the same global ranking for the explicit schemes also hold.

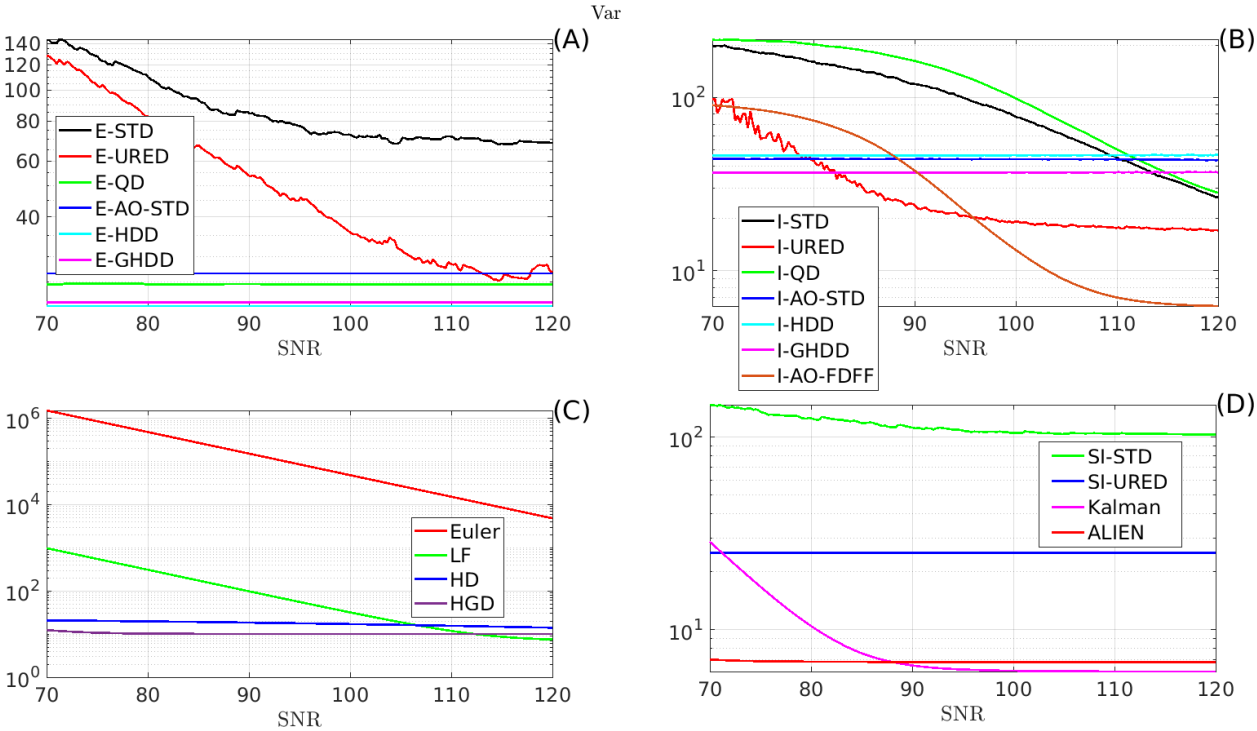

Figure 59: Variation for the third-order differentiation under different SNRs (h =10ms).

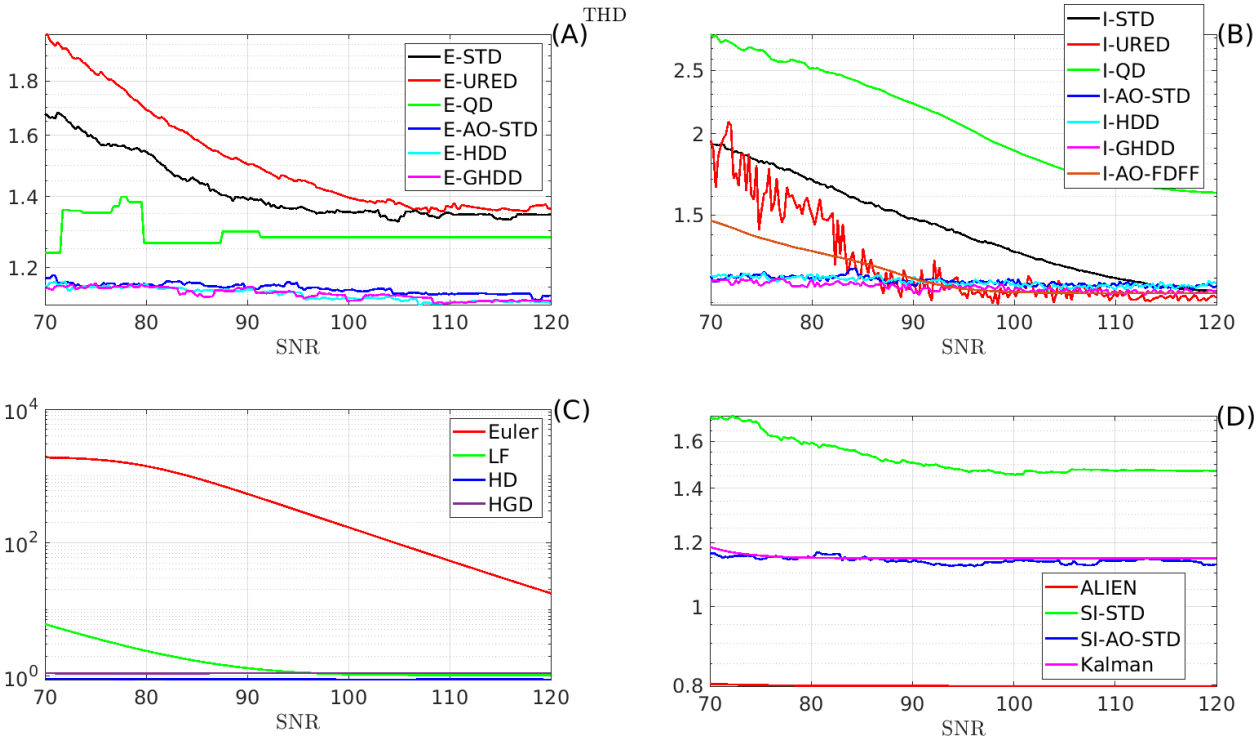

Figure 60: THD for the third-order differentiation under different SNRs (h =10ms).

Considering all the results in this section (Figs. 56 to 60), one can conclude that Kalman differentiator shows almost the best performances for calculating third-order differentiation of the input signal $\sin(t)$ in the presence of white noise. However, as it was investigated in Section 5.5, Kalman filter is too sensitive to initial error. Furthermore, comparison of the LF and the HGD shows that the cascade configuration of LF presents better results than those of the HGD.

5.7 Sensitivity of the exact differentiators to the parameters

To validate Remark 3, the output of the exact differentiators are presented for a relatively large differentiation gain ($L = F = \alpha = r = 100$) and shown in Fig. 61[1]. Other parameters are selected based on Table 2. To study the transient responses of the differentiators, all the initial conditions are set to zero. Thus, all these differentiators start from a non-zero initial condition for the input $\sin(t)$.

The most interesting result is that implicit schemes as well as HD and SI-AO-STD are insensitive to the gains and calculate the differentiation without chattering, see Fig. 61 (I), (J) and (K). Other differentiators including the explicit and the semi-implicit schemes are highly sensitive to the gains and present a significant amount of chattering (see Fig. 61 (C), (D), (E), (F), (G) and (H)). Note that E-URED and VGED diverge from the actual differentiation (Fig. 61 (A) and (B)) for these oversized gains. It is also interesting to note

[1]The maximum tolerable gain for the discrete-time HGD with $h = 50$ms is $L = 20$. For higher-gains it leads to instability because all L, L^2, L^3, L^4 appear in (16). Hence $L = 20$ is considered for the HGD

that in Fig. 61 (K) all the curves are identical excepted for the transient of I-AO-STD, I-HDD and I-GHDD. It means that all the implicit methods converge to the real differentiation after a finite-time which validates Corollary 1.

Note that the I-AO-STD, I-HDD and I-GHDD show significant amount of transient errors compared to that of other implicit schemes. This is also the case of E-AO-STD (Fig. 61 (G)). From Fig. 61 (K) it seems that AO-STD possesses a quite good chattering behavior and gain insensitivity, however at the price of possible overshoot during the transient period for too large gains (no overshoot is visible in Fig. 16 (A) and (B) for small gains). It was also concluded in Section 5.5 that a higher-order differentiator has more initial conditions (internal state variables) than lower-order differentiators. Therefore, a higher-order differentiator may need larger transient time which leads to the deterioration of the transient response. This may affect the closed-loop behavior when such algorithms are used for control purposes.

Among explicit methods the HD presents the best responses with the smallest chattering. Moreover the parameters of this differentiator are specifically designed for the discrete-time implementation which leads to a good transient response (see Section 3.2).

Remark 13 *If the differentiation gains are selected according to Corollary 2, the implicit methods will be exact and insensitive to the gains for any input during the sliding-phase. Since the differentiators cannot distinguish between the signal $f_0(t)$ and its noisy counterpart $f(t) = f_0(t) + n(t)$, the differentiators will be exact on the noisy input as well. To solve this problem,* (91b) *is taken into consideration to cancel the exactness of the I-STD on the high-frequency components of the input (with this assumption that the noise has higher frequency components than the signal).*

Remark 14 *From Fig. 61 (G), it can be seen that HD is also gain insensitive.*

Remark 15 *Fig. 61 shows that with increasing the gain, unlike HD, both E-AO-STD and the I-AO-STD may present overshoots. This is a very important observation. The reason is that the parameters λ_i which are shown in Table 2 are obtained for the continuous-time AO-STD. These parameters should be re-tuned for the discrete-time differentiators such as E-AO-STD and the I-AO-STD. In fact, the parameters of the HD are specifically designed for the discrete-time case which led to a better transient responses. Many studies use the parameters λ_i which are shown in Table 2. These parameters are for the continuous-time system, which should be re-tuned for the discrete-time differentiators to obtain a better transient response.*

An interesting observation from Table 15 is that for this case, where over-sized gain are considered for the differentiators, the behavior of the I-URED is the same as the I-STD. It means that for over-size-gains, the effect of the high-degree terms are negligible.

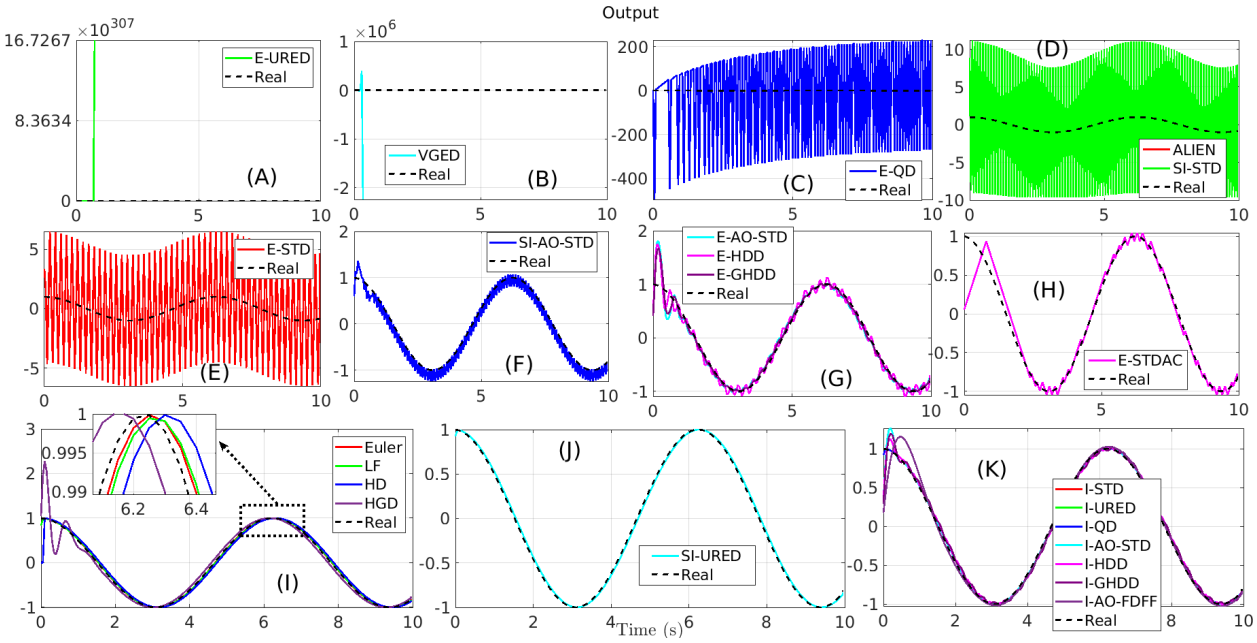

Figure 61: output of the differentiators with an oversized differentiation gains $L = F = \alpha = r = 100$ under a noise-free condition and $h = 50$ms.

Table 15: Results under oversized gains.

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0012	0.0273	0.0250	6.1735	4.4747	1.00 β
LF	0.0019	0.0399	0.1658	6.3099	4.3449	1.47 β
E-STD	0.2754	5.0001	5.6819	1097.3340	22.1108	1.64 β
I-STD	0.0012	0.0273	0.0250	6.1735	4.4747	1.93 β
SI-STD	0.6664	8.5677	10.2410	3756.0791	48.2453	1.64 β
E-URED	NaN	NaN	NaN	NaN	NaN	1.66 β
I-URED	0.0012	0.0273	0.0250	6.1735	4.4747	1.68 β
E-QD	17.0110	251.4740	495.9950	77725.0000	974.9762	2.08 β
I-QD	0.0015	0.0326	0.0830	6.2328	4.4126	2.45 β
ALIEN	NaN	NaN	NaN	NaN	NaN	63.07 β
HD	0.0079	0.1524	0.9988	7.1409	3.8474	7.35 β
E-AO-STD	0.0106	0.2261	0.8287	10.2872	4.6651	2.73 β
I-AO-STD	0.0045	0.0822	0.6576	7.3598	4.0787	7.19 β
SI-AO-STD	0.0118	0.2003	0.4056	50.9650	4.9360	3.57 β
E-HDD	0.0098	0.2024	0.7763	17.0832	4.6725	3.80 β
E-GHDD	0.0086	0.1821	0.7005	9.1418	4.4627	4.69 β
I-HDD	0.0046	0.0839	0.6624	7.6049	4.0259	33.48 β
I-GHDD	0.0052	0.0980	0.7122	7.7234	3.9808	22.49 β
VGED	NaN	NaN	NaN	NaN	NaN	16.06 β
SI-URED	0.0016	0.0360	0.0732	6.2289	4.4173	2.31 β
E-STDAC	0.0120	0.2556	0.9438	11.0000	3.2510	2.57 β
I-FDFF	0.0019	0.0399	0.1658	6.3099	4.3449	5.22 β
I-AO-FDFF	0.0019	0.0399	0.1702	6.3138	4.3432	13.47 β
Kalman	0.0810	1.8049	2.5150	15.5229	2.5826	6.95 β
HGD	0.0097	0.1948	1.0408	8.6577	5.6419	1.87 β

SNR= ∞, h=50ms. NaN stands for Not a Number and is achieved as the results of $0 \times \infty$, $\frac{\infty}{\infty}$, $\frac{0}{0}$. $\beta = 0.1660$ms. Performance: red<black<blue.

5.8 Sensitivity of the implicit method to the accuracy of the solver

The purpose of this section is to analyse the effect of the solver's accuracy on the differentiators' performances. As mentioned in Sections 3.3.3 to 3.3.6, I-AO-STD, I-HDD, I-GHDD and I-URED require an iterative (Newton-Raphson like) algorithm to solve a polynomial equation at each time-step (see (68), (73), (79), (83), (101) and (102)). So far, a built-in MATLAB function `roots` is utilized to solve these polynomial equations

in the foregoing simulations. Due to resource limitations in online applications, a suitable solver with appropriate accuracy should be selected to find a proper solution during the sampling interval. Therefore, this section is going to answer the following two questions:

- Which type of solver requires the minimum amount of calculation resources to solve the polynomial equation?
- How to select the accuracy of a solver to provide a balance between the performance and the required calculation resources?

As it is mentioned in Sections 3.3.4 to 3.3.6, implicit discretizations of AO-STD, HDD and GHDD lead to $(n+1)$th order polynomial equations, where n is the order of the differentiator. In this section, a third-order I-AO-STD is considered which leads to the following fourth-order polynomial equation:

$$x_k^4 + a_3 x_k^3 + a_2 x_k^2 + a_1 x_k + a_0 = 0. \tag{164}$$

According to Remark 4, the parameters a_i are such that this equation always has a unique real solution. In this section, few different solvers are introduced and the behavior of the I-AO-STD in the presence of these solvers is investigated.

5.8.1 Newton's method

Newton's method is a well-known iterative method to find the solutions of nonlinear continuous equations of the form of $p(x) = 0$ [73]. The formula of this method is:

$$x_n = x_{n-1} - \frac{p(x_{n-1})}{\dot{p}(x_{n-1})}, \tag{165}$$

where x_{n-1} and x_n are the previous and current guesses of the solution, $p(\cdot)$ and $\dot{p}(\cdot)$ are the function and its time derivative. In this case, $p(\cdot)$ denotes the polynomial which leads to the solution of the generalized equation. In this work, the solution of the previous step is always selected as the initial guess of the current step.

5.8.2 Householder's method

Householder's method is another iterative method for finding the real solutions of a nonlinear equation [74], which reads as:

$$x_n = x_{n-1} + d\frac{\left(\frac{1}{p(x_{n-1})}\right)^{(d-1)}}{\left(\frac{1}{p(x_{n-1})}\right)^{(d)}}, \tag{166}$$

where $p(\cdot)$ is a $d+1$ times continuously differentiable function. For $d = 2$, Householder's method leads to the Halley's approach which is used in Section 5.8.5.

5.8.3 Bairstow's method

Bairstow's method can be used to find all the roots (including the complex roots) of a real polynomial equation [75]. This method uses the Newton's method to adjust the parameters b_0 and b_1 of a quadratic polynomial $x^2 + b_1 x + b_0$, such that its roots are also the roots of the main polynomial. Subsequently, these roots will be eliminated from the main polynomial by it by the quadratic one. This procedure will be repeated until the main polynomial turns into a quadratic or linear polynomial.

5.8.4 MATLAB solver

MATLAB provides a function to calculate the roots of a polynomial equation. Here, MATLAB command `x=roots(P)` is used as a black box to calculate the roots, where `P` is the vector of the polynomial equation and `x` is a vector contains all the corresponding solutions.

5.8.5 Numerical simulations with different solvers

The effect of the solvers on the performance of a third-order I-AO-STD is investigated in this section. The parameter of this I-AO-STD is tuned such that it shows the smallest $\bar{L}_2$ norm. In these simulations, initial guess at each time step is set to the solution of the previous step, except for the first step where it is set to zero. Note that the simulations are performed for the noise-free condition and $h = 50$ms.

The effect of the Newton, Halley and Bairstow methods are presented in Tables 16 to 18, respectively. The parameters in the tables are defined as follows:

- **MNI:** Maximum number of iterations along all the sampling intervals. The processor must be able to perform these iterations at each time step, and these operations must take less than the sampling time h.
- **TNI:** Total number of iterations for the whole simulation time. This parameter shows how much a solver uses the calculation resources.
- **Acc:** Accuracy of a solution x_0 is defined as $\text{Acc}(x_0) = |p(x_0)|$, where $p(\cdot)$ is the polynomial. Smaller acc indicates better accuracy. For an exact solution one has Acc=0. The term MATLAB indicates the zero accuracy which is obtained by the MATLAB built-in function `roots`.

From Tables 16 to 18, it can be seen that the maximum accuracies for the Newton, Halley and Bairstow methods, without a significant deterioration on the performances, are 10^{-5}, 10^{-5} and 10^{-3}, respectively. For these accuracies, Newton's and Halley's methods show smaller MNI and TNI than those of the Bairstow's method. The most important parameter for online applications is the MNI which indicates the maximum amount of the required calculation resources at each time-step. Hence, comparing Tables 16 and 17, one may conclude that the Halley's is the best option for the implementations. However, since Halley's method performs more arithmetic operations at each time-step, the Newton's solver may lead to a lighter calculation

burden. Note that for the aforementioned accuracy, Newton's method shows much smaller TNI than the Halley's one.

Table 16: Effect of the solver's accuracy on the I-AO-STD with Newton's solver.

Acc	MNI	TNI	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD
MATLAB	-	-	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-12}	10	633	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-9}	9	50	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-6}	8	323	0.0011	0.0250	0.0246	6.2187	4.4589
10^{-5}	7	209	0.0011	0.0250	0.0241	6.2559	4.4520
10^{-4}	10	135	0.0013	0.0280	0.0300	6.3267	4.4393
10^{-3}	12	61	0.0023	0.0512	0.0662	6.5071	4.4116
10^{-2}	19	39	0.0061	0.1359	0.1482	6.9253	4.3655
10^{-1}	25	33	0.0426	0.9465	1.2330	9.7736	4.1269
$h = 50$ms, SNR=∞, input signal: $f(t)$=sin(t)							

Table 17: Effect of the solver's accuracy on the I-AO-STD with Householder's (Halley's) solver.

Acc	MNI	TNI	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD
MATLAB	-	-	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-12}	30	4884	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-9}	20	2926	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-6}	10	1029	0.0011	0.0249	0.0245	6.2253	4.4591
10^{-5}	6	565	0.0011	0.0249	0.0241	6.2554	4.4520
10^{-4}	6	267	0.0013	0.0285	0.0300	6.3377	4.4349
10^{-3}	3	75	0.0020	0.0451	0.0650	6.4689	4.4278
10^{-2}	7	33	0.0078	0.1739	0.1778	7.0924	4.3316
10^{-1}	9	29	0.0432	0.9625	1.2839	9.9972	3.9741
$h = 50$ms, SNR=∞, input signal: $f(t)$=sin(t)							

Table 18: Effect of the solver's accuracy on the I-AO-STD with Bairstow's solver

Acc	MNI	TNI	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD
MATLAB	-	-	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-12}	15	2126	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-9}	15	2040	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-6}	14	1923	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-5}	13	1848	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-4}	12	1719	0.0011	0.0249	0.0246	6.2258	4.4559
10^{-3}	11	1509	0.0011	0.0249	0.0245	6.2400	4.4575
10^{-2}	8	1135	0.0018	0.0403	0.0483	6.4463	4.4213
10^{-1}	8	648	0.0140	0.3096	0.3384	7.8360	4.4368
$h = 50$ms, SNR=∞, input signal: $f(t)$=sin(t)							

So far, it is shown that the minimum amount of MNI without deteriorating the performances of the I-AO-STD is 7, 6 and 11 for Newton's, Halley's and Bairstow's methods, respectively. However, it is not mentioned how much calculation resources are required in each iteration. To clarify this ambiguity, numerical simulations are conducted to calculate the required time for solving a typical fifth-order polynomial equation. These simulations show that for a typical condition, Newton's, Halley's, Bairstow's and MATLAB (`roots` built-in function) methods require 93.5μs, 30.50μs, 217.2ms and 159.1μs, respectively. As it can be seen, the Bairstow's method needs much larger time to calculate the roots. Hence, it is recommended to avoid using it. Moreover, Halley's method shows the smallest calculation time. Therefore, it can be concluded that for this specific case, Halley's method can provide a more efficient implementation for online calculations of the implicit methods compared to the other methods.

6 General conclusions

The following conclusions and guidelines may be drawn:

- Explicit discretization of exact differentiators leads to the numerical chattering. On the other hand, implicit discretization can resolve this drawback, even for large enough sampling times. Generally speaking, explicit methods should be avoided.
- In general, I-AO-STD, I-HDD and I-GHDD present the best responses in the presence of white (see Section 5.2.1), sinusoidal (see Section 5.2.2) and bell-shaped (see Section 5.2.3) types of noise. I-AO-FDFF also presents appropriate responses. However, this differentiator has six parameters which makes its implementation less convenient.

- According to the results in Section 5.5, arbitrary-order differentiators such as I-AO-STD have more initial conditions than the others, e.g., I-STD, due to more integrators. Hence, a higher-order differentiator shows a longer transient response. On the other hand, a higher-order differentiator, e.g., I-AO-STD may present better steady-state performances compared to a lower-order one.

- In general, some implicit schemes (I-AO-STD, I-HDD and I-GHDD) require more calculation resources than their explicit counterparts. But, it was shown that the implicit methods can also be implemented in real-time. Semi-implicit schemes can be utilized in applications with limited resources to provide a compromise between the performance and the calculation time (compare E-AO-STD, I-AO-STD and SI-AO-STD in Tables 3, 4, 7 and 11).

- Throughout the manuscript, it was observed that the order of an arbitrary-order differentiator can affect the performances. It is clear that the discontinuous signum function appears in the output of the STD through an integrator. As the result the E-STD (unlike the implicit one) shows a significant amount of chattering (see Fig. 61 (E)). To solve this drawback, E-AO-STD is proposed in the literature to attenuate the chattering amplitude. However, as it was shown in Fig. 16 (A) and (C), these extra integrators can increase the phase-lag of the differentiator, which leads to larger $\bar{L}_2$ and $\tilde{L}_2$ criteria than those of the I-STD (see Table 4). On the other hand, as it is also shown in Corollary 1, the implicit method converges the real differentiation in finite-time, under some conditions. It means that the implicit method does not present phase-lag compared to its explicit counterpart.

- In order to calculate the ith order differentiation, the order of the AO-STD must be at least $n = i$. A higher-order AO-STD ($n - i > 0$) can provide extra integrators to attenuate the effect of the high-frequency noise. This extra number of integrators can also help the explicit methods to decrease the chattering amplitude caused by the signum function (note that implicit methods suppress the chattering in an intrinsic way without requiring extra integrators). However, it should be noted that an extra number of integration affects the exactness of the differentiators and increases the phase-lag of an AO-STD.

- Numerical simulations confirmed that implicit differentiators (I-STD, I-URED, I-AO-STD, I-HDD, I-GHDD, and I-AO-FDFF) drastically supersede the linear filters in the presence of noise (see Sections 5.2.1 to 5.2.3).

- AO-STD provides extra filtration by calculating and integrating the higher-order differentiation. As the result, AO-STD usually presents better steady-state performances compared to those of the STD (for instance see Figs. 49 to 53), both in explicit and implicit implementations.

- While Kalman method presents one of the best performances in steady-states (see Tables 5, 7, 11 and 14), it is too sensitive to the disturbances corresponding to the initial error (see Table 13 and

Figs. 47 and 49 to 53). Hence, Kalman differentiator may not be considered as an appropriate solutions in closed-loop control systems where the initial states of the system are unknown.

- There is a group of differentiators (LF, STD, URED, QD, VGED, STDAC, FDFF) which usually possess similar performances for $\bar{L}_2$ criteria in noisy conditions (Tables 5, 7, 9 and 11).[1]
- Newton and Halley's algorithms are suitable iterative schemes to solve the generalized equations for implicit methods.

From an engineering point of view, the most appropriate differentiator can be selected according to the following table.

[1]We have not been able to find clear criteria that could distinguish these algorithms one from each others. It is possible, however, that in other conditions of test, some of them may behave better than the others.

Table 19: A guidance to select the most appropriate differentiation method. The characters 2, ∞, V, T and C stand for $\bar{L}_2$, L_∞, VAR, THD and the calculation time, respectively. Colors blue and red show the best and the worst performances, respectively.

Method	noise-free	white noise	sinusoidal noise	bell-shaped noise	quantization
Euler	2 ∞ V C	2 ∞ V T C	2 ∞ V T C	2 ∞ V T C	2 ∞ V T C
LF	V C	C	C	C	C
E-STD	C	C	C	C	C
I-STD	2 V C	C	-	-	C
SI-STD	C	C	C	C	C
E-URED	C	C	-	-	C
I-URED	C	C	C	C	C
E-QD	V C	C	-	-	C
I-QD	V	-	-	-	-
ALIEN	T C	T C	T C	T C	V T C
HD	T C	-	T C	T	-
E-AO-STD	-	-	-	-	-
I-AO-STD	2 ∞ V C	2 ∞ V C	2 ∞ V C	2 ∞ V C	2 ∞ V C
SI-AO-STD	-	-	-	-	-
E-HDD	-	-	-	-	-
E-GHDD	-	-	-	-	-
I-HDD	V C	2 ∞ V C	2 ∞ V C	2 ∞ V C	2 ∞ V C
I-GHDD	V C	2 ∞ V C	2 ∞ C	2 ∞ V C	2 ∞ V C
VGED	C	-	-	-	-
SI-URED	V C	C	-	-	C
E-STDAC	C	-	V	-	C
I-FDFF	V	C	-	-	C
I-AO-FDFF	C	2 ∞ C	2 ∞ V C	C	2 ∞ C
Kalman*	-	2 ∞ V	2 ∞ V T	V	2 ∞ V
HGD	-	-	-	-	-
*Kalman presents one of the worst transients					

6.1 Future works

The conclusions drawn from this study are true only for the open-loop differentiation. The case of the closed-loop system will be treated in the future and could modify some of the conclusions and recommendations.

Hence, future research will probably deal with the implicit differentiators in the closed-loop control systems as well as their practical implementations on laboratory set-ups. Some other possible topics for future research are also listed as follows:

- A crucial property to transport the Lyapunov stability properties from the continuous-time system to the discretized system is that the continuous-time Lyapunov function has convex level sets. Such a Lyapunov function has been provided in the literature for the super-twisting algorithm, i.e., AO-STD with $n = 1$ [58]. Thus, generalization of such a Lyapunov function for $n > 1$ can be considered in future works.
- The accuracy of the implicit differentiators has only been studied for $n = 1$ in Lemma 4. Generalization and investigation of this accuracy for $n > 1$ may be done in future studies.
- As it was seen, implicit methods need an online iterative solver to solve a polynomial equation at each time step. More efficient solvers may be designed for this purpose.
- Investigation of the implicit differentiators based on the homogeneity theorems [42] may also be an interesting topic.
- In this study the parameters of the differentiators are tuned using a stochastic optimization algorithm. However, a more systematic way may be developed for the parameter tuning.

A Auxiliary technical result

Proposition 1 ([37]) *The following equations are equivalent one to each other:*

- $z - y = \mathrm{gsgn}\left(\mathrm{gsgn}(a, -z, b), x - z, \mathrm{gsgn}(c, -z, d)\right)$
- $z - y = \mathrm{gsat}\left(\mathrm{gsat}(a, -y, b), x - y, \mathrm{gsat}(c, -y, d)\right)$,

where, $a \leq b \leq c \leq d$ *and:*

$$\mathrm{gsat}(a, z, b) = \begin{cases} a & \text{if} \quad z < a \\ z & \text{if} \quad z \in [a, b] \\ b & \text{if} \quad z > b. \end{cases} \tag{167}$$

B Proofs

Proof of Lemma 1. For any $L > 0$, there are always parameters λ_0 and λ_1 to satisfy Lemma 1 [58]. For the sake of simplicity, the proof will be provided for $0 \leq L < \frac{1}{2}$, $L < \lambda_1 < 1 - L$ and $\lambda_0 = \sqrt{2\sqrt{2}}$. A Lyapunov function candidate $V(\cdot)$ with ellipsoidal level sets can be obtained by the equality $Q(V, \sigma) = 0$ as follows [76]:

$$Q(V, \sigma) = \sigma^T D(V^{-1}) P D(V^{-1}) \sigma - 1, \tag{168}$$

where $\sigma = [\sigma_0, \sigma_1]^T \in \mathbb{R}^2$, $D(V^{-1}) = \begin{bmatrix} V^{-2} & 0 \\ 0 & V^{-1} \end{bmatrix}$, $P = \begin{bmatrix} 1+\varepsilon & -1 \\ -1 & 1 \end{bmatrix} > 0$, $\varepsilon \in \mathbb{R} > 0$. According to [76], the following inequality ensures the positive definiteness and well-posedness of the Lyapunov candidate:

$$PG_D + G_D^T P > 0, \tag{169}$$

where $G_D = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$. Equation (169) holds for $\varepsilon > 1/8$. The partial derivative of the Lyapunov function can be calculated as follows:

$$\frac{\partial V}{\partial \sigma} = \frac{2V\sigma^T D(V^{-1})PD(V^{-1})}{\sigma^T D(V^{-1})(PG_D + G_D^T P)D(V^{-1})\sigma}. \tag{170}$$

Transformation map of the STD (84) can be formulated as follows:

$$\dot{\sigma} \in F(\sigma) = \begin{bmatrix} -\lambda_0 \sqrt{|\sigma_0|}\,\mathrm{sgn}(\sigma_0) + \sigma_1 \\ -\lambda_1 \,\mathrm{sgn}(\sigma_0) + [-L, +L] \end{bmatrix}. \tag{171}$$

Using (170) and (171), partial derivative of the Lyapunov function along the trajectories of (171) is given by [58, 76]:

$$\sup_{\xi \in F(\sigma)} \frac{\partial V}{\partial \sigma}\xi = \frac{2V \sup\limits_{\xi \in F(\sigma)} \sigma^T D(V^{-1})PD(V^{-1})\xi}{\sigma^T D(V^{-1})(PG_D + G_D^T P)D(V^{-1})\sigma} \tag{172a}$$

$$= \frac{2 \sup\limits_{z \in F(D(V^{-1})\sigma)} \sigma^T D(V^{-1})Pz}{\sigma^T D(V^{-1})(PG_D + G_D^T P)D(V^{-1})\sigma}, \tag{172b}$$

where, according to (168), $\sigma^T D(V^{-1})PD(V^{-1})\sigma = 1$. From (169), the expression in (172b) is negative if:

$$\sup_{z \in F(y)} y^T Pz < 0, \quad \text{for} \quad y^T Py = 1, \tag{173}$$

where $y = [y_1, y_2]^T$. For $y_1 = 0$, $y^T Py = 1$. It implies $y_2^2 = 1$ and $F(y) = \begin{bmatrix} y_2 \\ [-\lambda_1, \lambda_1] + [-L, +L] \end{bmatrix}$. Thus, $\sup\limits_{z \in F(y)} y^T Pz = -1 + (\lambda_1 + L) < 0$. For the case $y_1 > 0$ (the case $y_1 < 0$ can be treated similarly), we have

$$\sup_{z \in F(y)} y^T Pz = \sup_{\tilde{L} \in [-L, L]} y^T P \begin{bmatrix} -\lambda_0 \sqrt{y_1} + y_2 \\ -\lambda_1 + \tilde{L} \end{bmatrix} = \tag{174}$$
$$\sup_{\tilde{L} \in [-L, L]} -\lambda_0(1+\varepsilon)y_1^{3/2} + \lambda_0 y_2 \sqrt{y_1} + (1+\varepsilon)y_1 y_2 - y_2^2 + (y_1 - y_2)(\lambda_1 - \tilde{L}).$$

The term $y^T Py = 1$ gives $y_2^2 - 2y_1 y_2 + (1+\varepsilon)y_1^2 = 1$ and

$$\begin{aligned} y_2 &= y_1 - \sqrt{1 - \varepsilon y_1^2} \quad \text{if} \quad y_1 - y_2 \geq 0 \\ y_2 &= y_1 + \sqrt{1 - \varepsilon y_1^2} \quad \text{if} \quad y_1 - y_2 < 0, \end{aligned} \tag{175}$$

with $y_1 \in \left(0, \frac{1}{\sqrt{\varepsilon}}\right]$. Thus, for $y_1 - y_2 \geq 0$ one has,

$$\sup_{z \in F(y)} y^T Pz \leq q_1(y_1) - (1 - \lambda_1 - L)\sqrt{1 - \varepsilon y_1^2}, \tag{176}$$

where

$$q_1(y_1) = -\lambda_0 \varepsilon y_1^{3/2} - \lambda_0 \sqrt{y_1}\sqrt{1-\varepsilon y_1^2} + 2\varepsilon y_1^2 + (1-\varepsilon) y_1 \sqrt{1-\varepsilon y_1^2} + \sqrt{1-\varepsilon y_1^2} - 1. \tag{177}$$

For $y_1 - y_2 < 0$:

$$\sup_{z \in F(y)} y^T P z \leq q_2(y_1) - (\lambda_1 - L)\sqrt{1-\varepsilon y_1^2}, \tag{178}$$

where

$$q_2(y_1) = -\lambda_0 \varepsilon y_1^{3/2} + \lambda_0 \sqrt{y_1}\sqrt{1-\varepsilon y_1^2} + 2\varepsilon y_1^2 - 1 + (\varepsilon - 1) y_1 \sqrt{1-\varepsilon y_1^2}, \tag{179}$$

If $\varepsilon = 1$ and $\lambda_0 = \sqrt{2\sqrt{2}}$, then

$$q_1(0) = 0, \qquad q_1(y_1) < 0 \qquad \text{for} \qquad y_1 \in (0, 1], \tag{180a}$$

$$q_2(1/\sqrt{2}) = 0, \qquad q_2(y_1) < 0 \qquad \text{for} \qquad y_1 \in (0, 1/\sqrt{2}) \cup (1/\sqrt{2}, 1], \tag{180b}$$

The detailed proofs of (180a) and (180b) are omitted for shortness (see [58]). Therefore, $V(\cdot)$ is a strict Lyapunov function for the continuous-time STD. The level sets of this function, $\Omega(r) = \{\sigma \in \mathbb{R}^2 | V(\sigma) \leq r$ $\}$, are as follows:

$$\Omega(r) = \{\sigma \in \mathbb{R}^2 | \sigma^T D(r^{-1}) P D(r^{-1}) \sigma \leq 1\}. \tag{181}$$

As a result, $\Omega(r)$ is an ellipsoid for any $r > 0$. ■

Proof of Lemma 2 Setting $\sigma_{i,k} = z_{i,k} - f_k^{(i)}$ into (41) gives

$$\begin{cases} \sigma_{i,k+1} = -h\lambda_i L^{\frac{i+1}{n+1}} \lceil \sigma_{0,k+1} \rfloor^{\frac{n-i}{n+1}} + h\sigma_{i+1,k+1} + \sigma_{i,k}, \quad i = 0, \ldots, n-1 & \text{(182a)} \\ \sigma_{n,k+1} \in -h\lambda_n L \operatorname{sgn}(\sigma_{0,k+1}) + \sigma_{n,k} + f_k^{(n+1)}. & \text{(182b)} \end{cases}$$

The term $f^{(n+1)}$ acts as a disturbance in (182). Assuming that $f^{(n+1)} = 0$, it is easy to show that once the system reaches its sliding surface $\sigma_k = \mathbf{0}$, it will remain on it thereafter, i.e., $\sigma_{k+1} = \mathbf{0}$. In other words, the sliding-surface behaves as an equilibrium point. ■

Proof of Lemma 3 Consider the level set of the implicit Lyapunov function (168) as follows:

$$\Omega(\lambda) = \{\sigma \in \mathbb{R}^n | V(\sigma) \leq \lambda\}, \tag{183}$$

and define $\delta : (0, \infty) \to (0, \infty)$ as $\delta(\lambda) = \inf_{\sigma \in \partial\Omega(\lambda)} W(\sigma)$, where $W(\cdot)$ is defined as:

$$\sup_{\xi \in F(\sigma)} \frac{\partial V}{\partial \sigma} \xi \leq -W(\sigma) \qquad \forall \sigma \in \mathbb{R}^n \backslash \{0\}, \tag{184}$$

and $\partial\Omega(\lambda)$ denotes the boundary of $\Omega(\lambda)$. Note that the set $\Omega(\lambda)$, as well as its boundary, are compact, because of continuity and radial unboundedness of the Lyapunov function. Moreover, the set-valued function $\Omega : [0, \infty) \to \mathbb{R}^n$ is continuous in the Hausdorff metric.

Thus, continuity/positivity of $W(\cdot)$ leads to continuity/positivity of the function δ. From (184), one has:

$$\sup_{\sigma\in\partial\Omega(\lambda)}\sup_{\xi\in F(\sigma)}\frac{\partial V}{\partial\sigma}\xi\leq-\sigma(\lambda)<0\quad\text{if}\quad\lambda>0. \tag{185}$$

If $\xi\in F(\sigma_{0,k+1})$, then for any $\sigma_{0,k}=\sigma_{0,k+1}-h\xi$, we have

$$\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}\xi=\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}\frac{\sigma_{0,k+1}-\sigma_{0,k}}{h}\leq-\delta(V(\sigma_{0,k+1})). \tag{186}$$

Since the level set $\Omega(V(\sigma_{0,k+1}))$ is convex, and $\sigma_{0,k+1}\in\partial(V(\sigma_{0,k+1}))$, then the last inequality implies the point $\sigma_{0,k}$ is separated from the convex compact set $\Omega(V(\sigma_{0,k+1}))$ by the tangential plane at the point x_{k+1}. Hence, the distance from $\sigma_{0,k}$ to $\Omega(V(\sigma_{0,k+1}))$ is greater than the distance from $\sigma_{0,k}$ to the tangential plane. The corresponding distance $|n^T(\sigma_{0,k}-\sigma_{0,k+1})|$ can be estimated as follows:

$$\begin{aligned}|n^T(\sigma_{0,k}-\sigma_{0,k+1})|&=\left|\frac{\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}}{||\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}||}(\sigma_{0,k}-\sigma_{0,k+1})\right|=\\&\frac{h|-\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}\frac{(\sigma_{0,k+1}-\sigma_{0,k})}{h}|}{||\frac{\partial V(\sigma_{0,k+1})}{\partial\sigma}||}\geq d(V(\sigma_{0,k+1})),\end{aligned} \tag{187}$$

where

$$d(\lambda)=h\delta(\lambda)\inf_{\sigma\in\partial\Omega(\lambda)}\left|\left|\frac{\partial V(\sigma)}{\partial\sigma}\right|\right|^{-1}>0\ \text{ if }\ \lambda>0. \tag{188}$$

Since $\left|\left|\frac{\partial V}{\partial\sigma}\right|\right|:\mathbb{R}^n\to\mathbb{R}$ is a continuous positive definite function, then $d:(0,\infty)\to(0,\infty)$ is continuous as well. The latter means

$$V(\sigma_{0,k+1})-V(\sigma_{0,k})<-\varepsilon_k, \tag{189}$$

where $\varepsilon_k>0$. Thus, it can be concluded that the sequence $V(\sigma_{0,\sigma})$ is monotone decreasing to zero as $k\to\infty$. Therefore, asymptotic convergence of the I-STD is ensured. ■

Proof of Corollary 1 From Lemma 3, the implicit discretization is asymptotically stable, i.e., $\sigma_{0,k+1}\to 0$ as $k\to\infty$. It means that there is a k_x such that for $k\geq k_x$, $|\sigma_{i,k}|<\epsilon$ for all $i=0,\ldots,n$. Choosing k_x large enough (but finite), it follows that $b_k=-\sum_{l=0}^{n}h^l\sigma_{l,k}\in[-Lh^{n+1}\lambda_n,Lh^{n+1}\lambda_n]$ for $k>k_x$. This corresponds to the Case 2 in Fig. 7. Substituting $\sigma_{i,k}=z_{i,k}-f_k^{(i)}$ into the equations of the second case in Fig. 7 gives

$$\begin{cases}\sigma_{i,k+1}=h\sigma_{i+1,k+1}+\sigma_{i,k},\quad i=0,\ldots,(n-1)\\ \sigma_{n,k+1}=\sigma_{n,k}+\frac{b_k}{h^n}\end{cases} \tag{190}$$

Substituting $b_k = -\sum_{l=0}^{n} h^l \sigma_{l,k}$ into (190) one has

$$\begin{cases} \sigma_{0,k+1} = 0 \\ \vdots \\ \sigma_{i,k+1} = \dfrac{-\sigma_{0,k} - \sigma_{1,k} - \cdots - h^{i-1}\sigma_{i-1,k}}{h^{i-1}} \\ \vdots \\ \sigma_{n,k+1} = \sigma_{n,k} + \dfrac{-\sigma_{0,k} - h\sigma_{1,k} - \cdots - h^n\sigma_{n,k}}{h^n} = \dfrac{-\sigma_{0,k} - \sigma_{1,k} - \cdots - h\sigma_{n-1,k}}{h^n} \end{cases} \tag{191}$$

It indicate that once the system arrives at the second case at the time step k, i.e., $b_k \in [-Lh^{n+1}\lambda_n, Lh^{n+1}\lambda_n]$, $\sigma_{0,k+1} = 0$ will be achieved. Here, it is reminded that under the condition (90), once the I-AO-STD reaches the Case 2, i.e., $b_k \in [-Lh^{n+1}\lambda_n, Lh^{n+1}\lambda_n]$, it will remain in this case thereafter. Hence, repeating the same procedure for the next time step gives

$$\begin{cases} \sigma_{0,k+2} = 0 \\ \sigma_{1,k+2} = 0 \\ \vdots \\ \sigma_{i,k+2} = \dfrac{-\sigma_{1,k+1} - \cdots - h^{i-1}\sigma_{i-1,k+1}}{h^{i-1}} \\ \vdots \\ \sigma_{n,k+2} = \sigma_{n,k+1} + \dfrac{-h\sigma_{1,k+1} - \cdots - h^n\sigma_{n,k+1}}{h^n} = \dfrac{-\sigma_{1,k+1} - \cdots - h\sigma_{n-1,k+1}}{h^n} \end{cases} \tag{192}$$

It shows that $\sigma_{1,k+2} = 0$ will also hold. Repeating the same procedure one can conclude that once the system arrives in Case 2, after $n+1$ steps the sliding-surface $\sigma_{i,k}, i = 0, \ldots, n$ is achieved which indicates the finite-time convergence of the AO-STD. ■

C Effect of the criteria on the parameter tuning

So far, the parameters of the differentiators have been tuned based on the $\bar{L}_2$ norm of the differentiation error (see Section 4). Other criteria will be considered in this section to investigate whether the previous results are still valid or not. The other criteria are listed as follow:

- $\tilde{\boldsymbol{L}}_{\boldsymbol{2}}$ **:** Comparing (145) and (146), it can be seen that $\tilde{L}_2$ is just the interpolation of $\bar{L}_2$ over a smaller sampling time. Therefore, tuning the parameters based on $\tilde{L}_2$, should result the same results as $\tilde{L}_2$.
- $\boldsymbol{L}_{\boldsymbol{\infty}}$ **:** Here, the parameters are obtained based on the L_∞ norm in the steady-state. The results are as follows:

Table 20: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min $J = 10000L_\infty$
Euler (3)	No parameter	6327.7426
LF (4)	c=8.3609	2277.7550
E-STD (21)	L=0.7961	1787.4148
I-STD (fig. 3)	L=0.7688	1679.6628
SI-STD (123)	L=0.7731	1958.0139
E-URED (26)	L=0.0761, μ=25.4435	1716.9083
I-URED (fig. 5)	L=0.1140, μ=23.6689	1627.0547
E-QD (25)	F=3.4035, α= 0.5940	2127.3314
I-QD (60)	F=4.8419, α=2.2309	2170.6543
ALIEN (14)	T=0.8166 , κ=1 , μ=5	2167.1280
HD (40)*	r=1.5305	2553.7412
E-AO-STD (22)**	L=4.8971	2024.6684
I-AO-STD (fig. 7) **	L=4.0099	1041.1867
SI-AO-STD (122)**	L=3.2248	1669.3520
E-HDD (24)**	L=4.9370	1706.9454
E-GHDD (27)**	L=4.9023	1682.3487
I-HDD (fig. 9)**	L=2.9907	960.4251
I-GHDD (fig. 10)**	L=3.0017	934.8679
VGED (35)	μ=4.1898, τ=0.2439, ω_c=13.4045, q=0.2843	1891.0830
SI-URED (139)	L=0.2223, μ=80.5437	2012.4431
E-STDAC (10)	α=0.3669, ϵ=0.0002	2001.1888
I-FDFF (Fig. 12)	ω_s=18.7669, ω_f=15.8499, ρ=20.5336, γ=0.0218	2057.3070
I-AO-FDFF (Fig. 12)	F=66.3231, ϵ=19.1486 ω_s=2.5079	1067.8845
	ω_f=29.7454, a_1=189.6309, ρ=140.5388	
Kalman (Section 3.5)	R=0.0007	1124.2598
HGD (16)	L=3.1446	2713.1916
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]).		
** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation		
Input signal: $\sin(t)$, Output: first-order differentiation		
Noise type: white, SNR=30dB, h = 50ms. Performance: red<black<blue		

Comparing Tables 3 and 20 one can see that the ranking which was obtained before, is still valid. In other words, the I-AO-STD, I-HDD, I-GHDD, I-AO-FDFF, and Kalman still present the best responses.

Furthermore, comparing Tables 5 and 21 one can conclude that the results which are obtained based on the $\bar{L}_2$ norm are still valid for the alternative criterion L_∞.

Table 21: Results for the first-order differentiation with white noise

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0401	0.6328	1.6678	156.1079	10.8251	1.00 β
LF	0.0116	0.2278	0.4422	30.9833	5.2808	1.56 β
E-STD	0.0094	0.1787	0.3868	24.9197	5.1148	2.01 β
I-STD	0.0088	0.1680	0.4037	24.2881	4.9385	1.93 β
SI-STD	0.0106	0.1958	0.4539	28.9841	5.3472	1.60 β
E-URED	0.0088	0.1717	0.4874	23.2035	5.0252	1.90 β
I-URED	0.0084	0.1627	0.4030	23.1537	4.8304	59.14 β
E-QD	0.0102	0.2127	0.4014	25.1210	4.6587	1.87 β
I-QD	0.0108	0.2171	0.5171	26.7401	4.7816	3.08 β
ALIEN	0.0053	0.1165	0.2167	7.8134	8.0307	30.42 β
HD	0.1258	1.9215	4.6755	28.8763	4.5124	12.06 β
E-AO-STD	0.0093	0.2025	0.2948	11.7062	4.7249	3.65 β
I-AO-STD	0.0049	0.1041	0.1662	9.5152	4.4505	61.25 β
SI-AO-STD	0.0077	0.1669	0.2531	10.8374	4.6043	4.42 β
E-HDD	0.0079	0.1707	0.2622	11.7385	4.6824	4.51 β
E-GHDD	0.0078	0.1682	0.2481	11.1058	4.6560	5.36 β
I-HDD	0.0045	0.0960	0.1415	8.7479	4.4055	58.51 β
I-GHDD	0.0043	0.0935	0.1417	8.4566	4.3993	56.22 β
VGED	0.0089	0.1891	0.4100	16.4619	5.0082	26.10 β
SI-URED	0.0095	0.2012	0.3226	14.2681	4.9851	1.87 β
E-STDAC	0.0091	0.2001	0.2498	11.5377	4.4822	2.37 β
I-FDFF	0.0102	0.2057	0.3633	26.1659	5.1150	2.49 β
I-AO-FDFF	0.0050	0.1068	0.1871	10.5364	4.4502	7.08 β
Kalman	0.0052	0.1124	0.1942	8.5897	4.3462	11.69 β
HGD	0.0098	0.2146	0.2713	10.0183	4.5816	2.22 β
h=50ms, and parameters are shown in Table 22. $\beta = 0.0720$ms. Performance: red<black<blue						

- **VAR:** Tuning the parameters based on VAR does not lead to the desired results since the optimization algorithm tries to reduce the differentiation gains in order to keep the variation as small as possible. In this case the gains tend to zero which leads to deterioration of the other criteria.

- **THD:** In addition to the $\bar{L}_2$ and L_∞ norms, we tried to consider the THD as the objective function to tune the parameters. The parameters obtained from the tuning procedure are presented in Table 22. One can see that, as before, ALIEN presents the smallest THD which comply with the previous results. Moreover, comparing Tables 5 and 23, it can be concluded that tuning the parameters based on THD criterion does not change the previous conclusions.

Table 22: Parameters of the differentiators obtained from the tuning procedure

Method	Parameters	Min J=THD
Euler (3)	No parameter	10.8251
LF (4)	c=3.1707	4.7232
E-STD (21)	L=0.6948	4.6953
I-STD (fig. 3)	L=0.6728	4.5925
SI-STD (123)	L=0.6771	4.8692
E-URED (26)	L=0.0048, μ=23.8513	4.2942
I-URED (fig. 5)	L=0.0223, μ=12.7608	4.3535
E-QD (25)	F=25.0466, α= 0.2201	3.6657
I-QD (60)	F=0.8075, α=46.2978	3.6045
ALIEN (14)	T=3.3439 , κ=4 , μ=1	2.4686
HD (40)*	r=1.0856	3.1495
E-AO-STD (22)**	L=0.6602	4.3328
I-AO-STD (fig. 7) **	L=1.0209	4.3035
SI-AO-STD (122)**	L=0.9748	4.4066
E-HDD (24)**	L=0.9629	4.3719
E-GHDD (27)**	L=0.9600	4.3412
I-HDD (fig. 9)**	L=0.9084	4.2937
I-GHDD (fig. 10)**	L=0.9627	4.2745
VGED (35)	μ=1.4532, τ=13.8745, ω_c=12.1723, q=1.5310	4.4262
SI-URED (139)	L=4.1955, μ=0.0753	4.5384
E-STDAC (10)	α=0.5113, ϵ=0.0010	4.3822
I-FDFF (Fig. 12)	ω_s=0.4156, ω_f=57.8406, ρ=3.7285, γ=0.3392	4.6704
I-AO-FDFF (Fig. 12)	F=53.1176, ϵ=13.0703 ω_s=1.5864	4.3572
	ω_f=45.0850, a_1=229.7572, ρ=116.2383	
Kalman (Section 3.5)	R=0.1391	4.2042
* Third-order HD which is utilized to calculate the first-order differentiation (see [8]).		
** Third-order AO-STD ($n = 3$) which is utilized to calculate the first-order differentiation		
Input signal: $\sin(t)$, Output: first-order differentiation		
Noise type: white, SNR=30dB, h = 50ms. Performance: red<black<blue		

Table 23: Results for the first-order differentiation with white noise

Method	$\bar{L}_2$	$\tilde{L}_2$	L_∞	VAR	THD%	Calculation time
Euler	0.0401	0.6328	1.6678	156.1079	10.8251	1.00 β
LF	0.0163	0.3604	0.4395	14.3201	4.7232	1.30 β
E-STD	0.0099	0.2059	0.4263	17.2416	4.6953	2.32 β
I-STD	0.0097	0.2026	0.3871	17.5601	4.5925	2.28 β
SI-STD	0.0106	0.2163	0.4307	18.9559	4.8692	1.62 β
E-URED	0.0219	0.4848	0.5967	12.0523	4.2942	1.84 β
I-URED	0.0185	0.4099	0.5172	12.3631	4.3535	60.74 β
E-QD	0.4010	8.9542	13.7873	89.3142	3.6657	1.85 β
I-QD	0.0236	0.5250	0.6935	7.2155	3.6045	3.22 β
ALIEN	0.0772	1.7196	1.6699	4.1630	2.4686	60.92 β
HD	0.0239	0.5284	0.9988	8.3733	3.1495	10.03 β
E-AO-STD	0.0256	0.5695	0.5315	9.0547	4.3328	3.80 β
I-AO-STD	0.0082	0.1834	0.2354	7.5913	4.3035	59.72 β
SI-AO-STD	0.0124	0.2768	0.2945	8.1481	4.4066	4.15 β
E-HDD	0.0130	0.2895	0.2975	8.0555	4.3719	4.30 β
E-GHDD	0.0136	0.3028	0.3081	8.0353	4.3412	5.27 β
I-HDD	0.0079	0.1767	0.2576	7.5213	4.2937	57.07 β
I-GHDD	0.0077	0.1707	0.2494	7.4934	4.2745	57.07 β
VGED	0.0138	0.3065	0.4103	11.3563	4.4262	26.30 β
SI-URED	0.0143	0.3176	0.3695	9.9007	4.5384	1.96 β
E-STDAC	0.0090	0.1992	0.2567	11.7444	4.3822	2.16 β
I-FDFF	0.0197	0.4349	0.5393	11.3966	4.6704	2.73 β
I-AO-FDFF	0.0113	0.2505	0.3184	8.2818	4.3572	7.05 β
Kalman	0.0185	0.4120	0.4176	8.4285	4.2042	9.64 β
HGD	0.0188	0.4208	0.4558	8.6387	4.2838	2.22 β

h=50ms, and parameters are shown in Table 22.

$\beta = 0.0870$ms. Performance: red<black<blue

D Analysis of the differentiators based on the homogeneity properties

The purpose of this section is to analyze the homogeneity properties of the super-twisting differentiator [1]. Consider the STD as follows:

$$\begin{cases} \dot{\sigma}_0(t) = -\lambda_0|\sigma_0|^{1/2}\,\mathrm{sgn}(\sigma_0) + \sigma_1 \\ \dot{\sigma}_1(t) = -\lambda_1\,\mathrm{sgn}(\sigma_0). \end{cases} \tag{193}$$

It is assumed that the second differentiation is zero, i.e., $\ddot{f} = 0$ (see (84)). In order to check the homogeneity of (193), the definition of weighted homogeneity is presented as Definitions 4 and 5.

Definition 4 *Positive numbers $r_i = 0, \ldots, n$ are called weights, and the vector of weights is defined as: $r = (r_0, \ldots, r_n)^T$.*

Definition 5 *The dilation matrix function is defined as $\Lambda_r(\varepsilon) = \mathrm{diag}\{\varepsilon^{r_i}\}, i = 0, \ldots, n$. for all $\sigma = [\sigma_0, \ldots, n] \in \mathbb{R}^n$, and for all $\varepsilon > 0$ one has $\Lambda_r(\varepsilon)\sigma = (\varepsilon^{r_0}\sigma_0, \ldots, \varepsilon^{r_n}\sigma_n)^T$.*

Definition 6 ***[63]*** *A function $g : \mathbb{R}^n \to \mathbb{R}$ is r-homogeneous with degree $v \in \mathbb{R}$ if for all $\sigma \in \mathbb{R}^n$ and for all $\varepsilon > 0$ one has:*

$$\varepsilon^{-\mu} g(\Lambda_r(\varepsilon)\sigma) = g(\sigma). \tag{194}$$

Definition 7 ***[63]*** *A vector field $f : \mathbb{R}^n \to \mathbb{R}^n$ is r-homogeneous with degree $v \in \mathbb{R}$ and $v \geq -\min(r_i)$ if for all $\sigma \in \mathbb{R}^n$, and for all $\varepsilon > 0$ one has:*

$$\varepsilon^{-v}\Lambda_r^{-1} f(\Lambda_r(\varepsilon)\sigma) = f(\sigma). \tag{195}$$

Since the transformation map of the STD (193) is a vector field, using Definition 7 it follows that:

$$\varepsilon^{-v}\begin{bmatrix} \varepsilon^{-r_0} & 0 \\ 0 & \varepsilon^{-r_1} \end{bmatrix}\begin{bmatrix} -\lambda_0|\varepsilon^{r_0}\sigma_0(t)|^{1/2}\,\mathrm{sgn}(\sigma_0(t)) + \varepsilon^{r_1} z_1(t) \\ -\lambda_1\,\mathrm{sgn}(\sigma_0(t)) \end{bmatrix} = \begin{bmatrix} -\lambda_0|\sigma(t)|^{1/2}\,\mathrm{sgn}(\sigma(t)) + z_1(t) \\ -\lambda_1\,\mathrm{sgn}(\sigma(t)) \end{bmatrix}. \tag{196}$$

It is clear that for $r_0 = 2, r_1 = 1, v = -1$, then (196) holds. Therefore, according to Definition 7, the continuous-time STD is homogeneous with degree $v = -1$.

Now, let us study the homogeneity degree of the STD in discrete-time form. In this case, the following definition called D_r-homogeneity is used to check the homogeneity:

Definition 8 ***[77]*** *Let Λ_ε^r be as before. A map $f : \mathbb{R}^n \to \mathbb{R}^n$, $f = [f_0, \ldots, f_n]$ is D_r-homogeneous of degree v if for each $i = 0, \ldots, n$, one has $f_i(\Lambda_\varepsilon^r\sigma) = \varepsilon^{r_i v} f_i(\sigma)$, for all $\sigma \in \mathbb{R}^n$, for all $\varepsilon \in \mathbb{R} > 0$, and some $v \in \mathbb{R} > 0$.*

[1] As a base reference, one may refer to [42] for further analysis of homogeneous systems.

Explicit transformation map of STD is as follows:

$$\begin{bmatrix}\sigma_{0,k+1}\\ \sigma_{1,k+1}\end{bmatrix} = \begin{bmatrix}-h\lambda_0|\sigma_{0,k}|^{1/2}\operatorname{sgn}(\sigma_{0,k}) + h\sigma_{1,k} + \sigma_{0,k}\\ -h\lambda_1\operatorname{sgn}(\sigma_{0,k}) + \sigma_{1,k}\end{bmatrix}. \tag{197}$$

It is easy to show that explicit discretization of STD is not D_r-homogeneous. Repeating this procedure for implicit discretization also leads to an identical result. To resolve this problem, an approximation of D_r-homogeneity is utilized as follows:

Definition 9 ***[78]*** *for a constant $\varepsilon_0 \in \mathbb{R} \geq 0 \cup \{\infty\}$, the transformation map $f : \mathbb{R}^n \to \mathbb{R}^n$ is said to be D_r-homogeneous of degree $v \in \mathbb{R} > 0$ in the $(\varepsilon_0, H) -$ limit, where $H : \mathbb{R}^n \to \mathbb{R}^n$ is some D_r-homogeneous map of degree v, if*

$$\lim_{\varepsilon\to\varepsilon_0}(\Lambda_\varepsilon^{-vr} f(\Lambda_\varepsilon^r \sigma) - H(\sigma)) = 0. \tag{198}$$

Using Definition 9, it is also easy to show that STD is not approximately D_r-homogeneous. As the result, it is not possible to use the theorems related to homogeneity for the discrete-time super-twisting algorithm.

E Differentiation toolbox

A toolbox is developed in MATLAB during this research to conduct all the necessary numerical simulations. Some of the features of the toolbox are listed as follows:

- All the differentiators which are considered in this manuscript are implemented in the toolbox including Euler, LF, E-STD, I-STD, E-QD, I-QD, E-URED, I-URED, E-AO-STD, I-AO-STD, ALIEN, HD, E-HDD, I-HDD, E-GHDD, I-GHDD, VGED, STDAC, I-FDFF, I-AO-FDFF, Kalman. Moreover, both semi-implicit discretization schemes (see Sections 3.4.1 and 3.4.2) are also considered in the toolbox.
- Calculate and draw the responses of the differentiators for almost any type of input signal and noise.
- Calculate and draw all the objective functions which are introduced in Section 4 for different SNR, sampling times, input frequencies, etc.
- A heuristic parameter tuning algorithm to tune all the parameters.

The toolbox contains many functions and scripts. But, the user only needs to consider two scripts named `main.m` and `parameters.m`. To run the simulation, just run the script `main.m`, and to customize the simulation one needs to edit the script `parameters.m`. The following parameters/variables in the script `parameters.m` can be customized:

- `t_f`: This variable determines the simulation time in seconds.

- `sample_length`: For `sample_length=1`, the toolbox will only provide the output of the differentiators as well as the corresponding performance functions in the command window. For `sample_length>1` the objective functions will be drawn with respect to the parameter which is specified by `changing_parameter`.
- `changing_parameter`: Depending on the value of this variable, the toolbox will draw the objective functions with respect to h, input frequency (ω), ω/h, h/ω, the amplitude of noise, and the frequency of noise. for `changing_parameter='param'`, the toolbox will calculate the optimal parameters.
- `diff_order`: This variable determines the desired differentiation order. All the objective functions will be drawn based on the specified order.
- `noise_type`: Determines the type of the noise, e.g., white or sinusoidal or bell-shaped.
- `SNR`: This variable indicates the SNR
- `Wn_vector`: Frequency of the input noise (for sinusoidal noise only).
- `An_vector`: Amplitude of the input noise (for sinusoidal noise only).
- `amp`: Amplitude of the input signal.
- `phi`: Phase of the input signal.
- `h_vector`: Sampling time.
- `w_vector`: Frequency of the input signal
- `ssp`: `ssp=1` and `ssp=0` determine the transient or steady-state performances, respectively.
- `hc`: Frequency of the interpolation.
- `k1, k2, k3, k4, k5`: Weighting functions for the global objective function.
- `AO_order`: The order of the AO-STDs

To customize the input signal and noise, the user may need to edit the function `signal_generator`. In addition to the plots, this toolbox is compatible with LaTeX and provides all the results as LaTeX tables in the command window. Overview of the toolbox is presented in Fig. 62.

The toolbox is available upon request to the first author: Mohammad Rasool Mojallizadeh (email: mohammad-rasool.mojallizadeh@inria.fr)

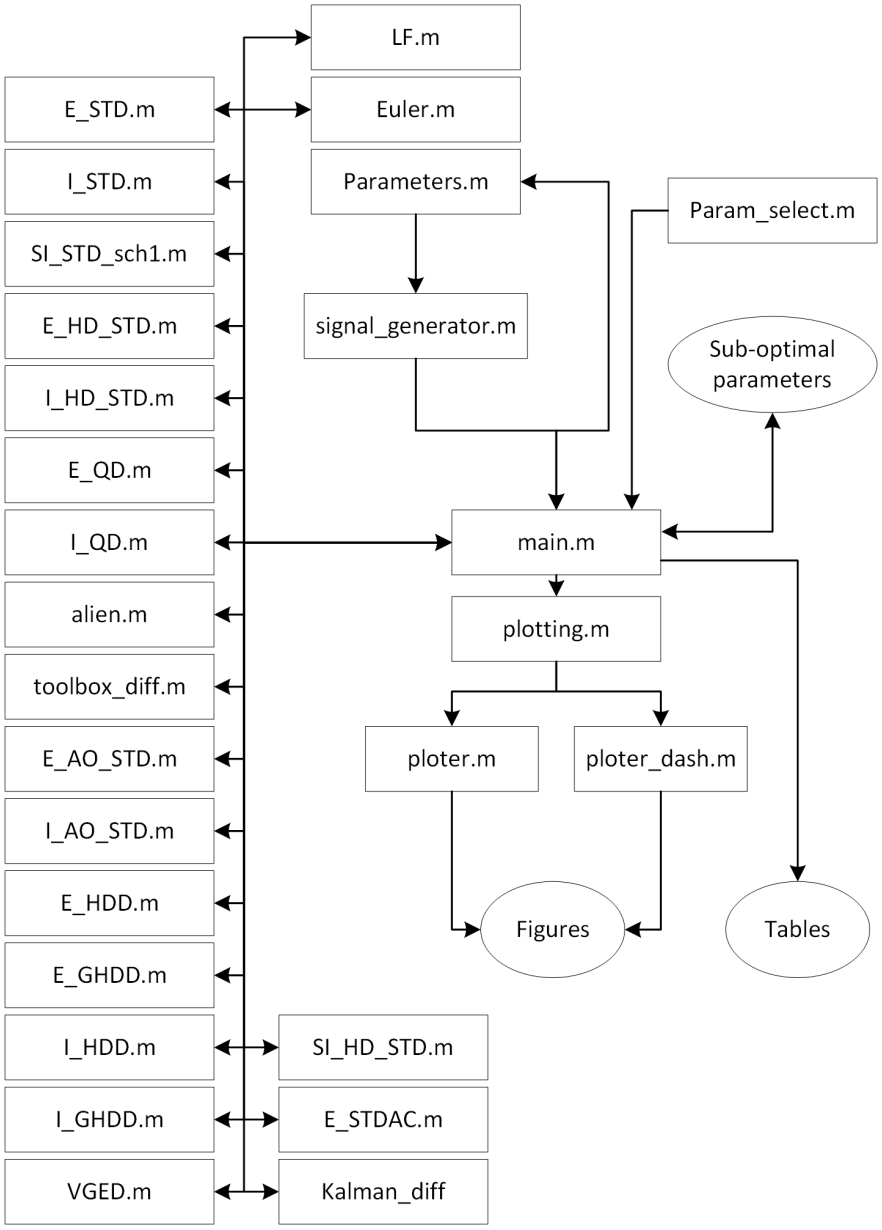

Figure 62: Overview of the MATLAB toolbox

F Quantization error

Physical quantities such as position, velocity, acceleration, temperature, pressure, etc. are analog. In order to measure and store these quantities as data in computers, we need to utilize a quantizer.

For example, consider a position sensor with the following specifications:

Output $\in$ [0,4] meters
Buffer size: one byte (eight bits)

Since the output range of the sensor is four meters and the buffer has 8 bits, the resolution of the sensor will be $\frac{4}{2^8} = 1.56$ centimeters. It indicates that the quantized data cannot detect the changes smaller than 1.56 cm.

Quantization and discretization are similar concepts which are used to digitize the time (x-axis) and the amplitude (y-axis), respectively. The effect of these digitizing schemes is shown in Fig. 63.

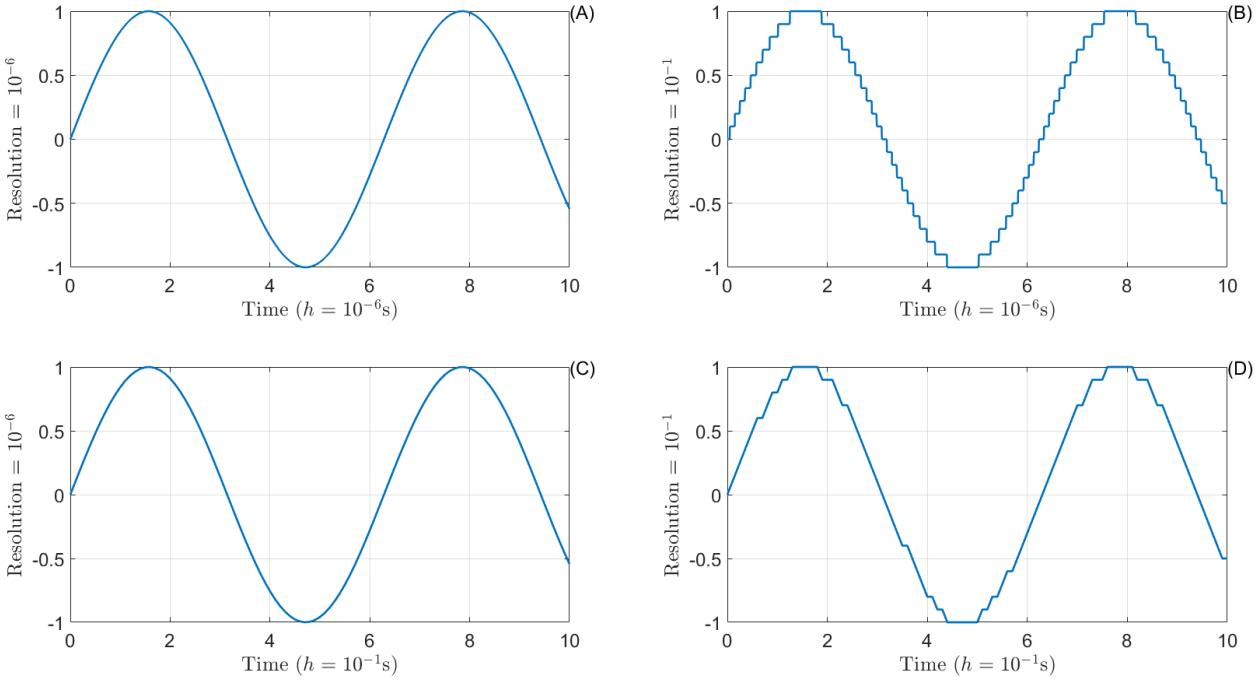

Figure 63: Digitizing $\sin(t)$

MATLAB command `fq=quant(f,q)` can be used to quantify a signal, where `f` is an analog signal, `q` is the resolution of the measurements, and `fq` is the quantized output.

In closed-loop applications (with or without differentiation), this quantization can cause chattering. The reason is that (as it can be seen from Fig. 63 (B)), for a small h, the quantization behaves as set-valued function. In fact, the output is set-valued in several points. Perhaps, this quantization error can be modeled as the summation of several set-valued-sign functions and then using an implicit quantization to remove the chattering corresponds to the quantization.

Acknowledgement

This work was supported by the project Digitslid (Différentiateurs et commandes homogènes par modes glissants multivalués en temps discret: l'approche implicite), ANR-18-CE40-0008-01

We would like to thank Dr. Alexandre Rocca for his help.

References

[1] M. Kumar, T. K. Rawat, A. Jain, A. A. Singh, and A. Mittal, "Design of digital differentiators using interior search algorithm," *Procedia Computer Science*, vol. 57, pp. 368–376, 2015. 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015).

[2] N. Q. Ngo, "A new approach for the design of wideband digital integrator and differentiator," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 9, pp. 936–940, 2006.

[3] H. K. Khalil, *Nonlinear systems*. Pearson Education, 2015.

[4] M. Mboup, C. Join, and M. Fliess, "Numerical differentiation with annihilators in noisy environment," *Numerical Algorithms*, vol. 50, no. 4, pp. 439–467, 2009.

[5] R. Ushirobira, "Algebraic differentiators through orthogonal polynomials series expansions," *International Journal of Control*, vol. 91, no. 9, pp. 2082–2089, 2018.

[6] S. Koch, M. Reichhartinger, M. Horn, and L. Fridman, "Discrete-time implementation of homogeneous differentiators," *IEEE Transactions on Automatic Control*, vol. In press, pp. 1–1, 2019.

[7] M. Reichhartinger, S. Spurgeon, M. Forstinger, and M. Wipfler, "A robust exact differentiator toolbox for MatLab/Simulink," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1711–1716, 2017.

[8] M. Reichhartinger, S. Koch, H. Niederwieser, and S. K. Spurgeon, "The robust exact differentiator toolbox: Improved discrete-time realization," in *2018 15th International Workshop on Variable Structure Systems (VSS)*, pp. 1–6, 2018.

[9] A. Levant, "Filtering differentiators and observers," in *2018 15th International Workshop on Variable Structure Systems (VSS)*, pp. 174–179, 2018.

[10] A. Levant, "Homogeneous filtering and differentiation based on sliding modes," in *2019 IEEE 58th Conference on Decision and Control (CDC), Nice, France*, pp. 6013–6018, 2019.

[11] D. V. Efimov and L. Fridman, "A hybrid robust non-homogeneous finite-time differentiator," *IEEE Transactions on Automatic Control*, vol. 56, no. 5, pp. 1213–1219, 2011.

[12] A. Levant, "Robust exact differentiation via sliding mode technique," *Automatica*, vol. 34, no. 3, pp. 379–384, 1998.

[13] A. Levant, "Higher order sliding: differentiation and black-box control," in *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 2, pp. 1703–1708, 2000.

[14] A. Levant, "Higher-order sliding modes, differentiation and output-feedback control," *International Journal of Control*, vol. 76, no. 9-10, pp. 924–941, 2003.

[15] A. Levant, "Higher order sliding modes and arbitrary-order exact robust differentiation," in *2001 European Control Conference (ECC)*, pp. 996–1001, 2001.

[16] R. Kikuuwe, R. Pasaribu, and G. Byun, "A first-order differentiator with first-order sliding mode filtering," *IFAC-PapersOnLine*, vol. 52, no. 16, pp. 771–776, 2019.

[17] G. Byun and R. Kikuuwe, "An improved sliding mode differentiator combined with sliding mode filter for estimating first and second-order derivatives of noisy signals," *Int. J. Control Autom. Syst.*, 2020.

[18] P. Li, G. Pin, G. Fedele, and T. Parisini, "Non-asymptotic numerical differentiation: a kernel-based approach," *International Journal of Control*, vol. 91, no. 9, pp. 2090–2099, 2018.

[19] L. K. Vasiljevic and H. K. Khalil, "Error bounds in differentiation of noisy signals by high-gain observers," *Systems & Control Letters*, vol. 57, no. 10, pp. 856–862, 2008.

[20] Y. Wang, G. Zheng, D. Efimov, and W. Perruquetti, "Differentiator application in altitude control for an indoor blimp robot," *International Journal of Control*, vol. 91, no. 9, pp. 2121–2130, 2018.

[21] M. Mboup, C. Join, M. Fliess, Y. Wang, G. Zheng, D. Efimov, and W. Perruquetti, "Comments on 'differentiator application in altitude control for an indoor blimp robot'," *International Journal of Control*, vol. 93, no. 5, pp. 1218–1219, 2018.

[22] H. Ahmed, H. Ríos, B. Ayalew, and Y. Wang, "Second-order sliding-mode differentiators: an experimental comparative analysis using Van der Pol oscillator," *International Journal of Control*, vol. 91, no. 9, pp. 2100–2112, 2018.

[23] W. Hongwei and W. Heping, "A comparison study of advanced tracking differentiator design techniques," *Procedia Engineering*, vol. 99, pp. 1005 – 1013, 2015. 2014 Asia-Pacific International Symposium on Aerospace Technology, APISAT2014 September 24-26, 2014 Shanghai, China.

[24] R. H. Jaafar and S. S. Saab, "Approximate differentiator with varying bandwidth for control tracking applications," *IEEE Control Systems Letters*, pp. 1–1, 2020.

[25] J.-J. E. Slotine, J. K. Hedrick, and E. A. Misawa, "On Sliding Observers for Nonlinear Systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 109, no. 3, pp. 245–252, 1987.

[26] A. Levant, M. Livne, and X. Yu, "Sliding-mode-based differentiation and its application," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1699 – 1704, 2017.

[27] A. Levant and X. Yu, "Sliding-mode-based differentiation and filtering," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 3061–3067, 2018.

[28] D. Deepika, S. Narayan, and S. Kaur, "Synthesis of a new robust exponential sliding mode differentiator with its observer applications," *Asian Journal of Control*, vol. 21, no. 1, pp. 387–396, 2019.

[29] E. Cruz-Zavala and J. A. Moreno, "Levant's arbitrary-order exact differentiator: A Lyapunov approach," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 3034–3039, 2019.

[30] H. Saadaoui, M. Djemaï, N. Manamanni, T. Floquet, and J.-P. Barbot, "Exact differentiation via sliding mode observer for switched systems," *IFAC Proceedings Volumes*, vol. 39, no. 5, pp. 124 – 129, 2006.

[31] A. Levant and M. Livne, "Robust exact filtering differentiators," *European Journal of Control*, vol. In Press, 2019.

[32] E. Cruz-Zavala and J. A. Moreno, "Lyapunov functions for continuous and discontinuous differentiators," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 660–665, 2016.

[33] M. Livne and A. Levant, "Proper discretization of homogeneous differentiators," *Automatica*, vol. 50, no. 8, pp. 2007 – 2014, 2014.

[34] E. Cruz-Zavala, J. A. Moreno, and L. M. Fridman, "Uniform robust exact differentiator," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2727–2733, 2011.

[35] W. Alam, Q. Khan, R. A. Riaz, and R. Akmeliawati, "Glucose–insulin stabilization in type-1 diabetic patient: A uniform exact differentiator–based robust integral sliding mode control approach," *International Journal of Distributed Sensor Networks*, vol. 15, no. 3, pp. 1–10, 2019.

[36] T. Emaru and T. Tsuchiya, "Research on estimating the smoothed value and the differential value of the distance measured by an ultrasonic wave sensor," in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, pp. 1291–1297, 2000.

[37] S. Jin, R. Kikuuwe, and M. Yamamoto, "Real-time quadratic sliding mode filter for removing noise," *Advanced Robotics*, vol. 26, no. 8-9, pp. 877–896, 2012.

[38] Z. Lv, S. Jin, X. Xiong, and J. Yu, "A new quick-response sliding mode tracking differentiator with its chattering-free discrete-time implementation," *IEEE Access*, vol. 7, pp. 130236–130245, 2019.

[39] W. Perruquetti, T. Floquet, and E. Moulay, "Finite-time observers: Application to secure communication," *IEEE Transactions on Automatic Control*, vol. 53, no. 1, pp. 356–360, 2008.

[40] A. Levant and M. Livne, "Weighted homogeneity and robustness of sliding mode control," *Automatica*, vol. 72, pp. 186–193, 2016.

[41] S. Koch and M. Reichhartinger, "Discrete-time equivalent homogeneous differentiators," in *2018 15th International Workshop on Variable Structure Systems (VSS)*, pp. 354–359, 2018.

[42] A. Polyakov, *Generalized Homogeneity in Systems and Control*. Springer, 2020.

[43] M. Reichhartinger and S. Spurgeon, "An arbitrary-order differentiator design paradigm with adaptive gains," *International Journal of Control*, vol. 91, no. 9, pp. 2028–2042, 2018.

[44] C. Vazquez, S. Aranovskiy, L. B. Freidovich, and L. M. Fridman, "Time-varying gain differentiator: A mobile hydraulic system case study," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 5, pp. 1740–1750, 2016.

[45] J. A. Moreno, "Exact differentiator with varying gains," *International Journal of Control*, vol. 91, no. 9, pp. 1983–1993, 2018.

[46] M. Ghanes, J. P. Barbot, L. Fridman, and A. Levant, "A novel differentiator: A compromise between super twisting and linear algorithms," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, Australia*, pp. 5415–5419, 2017.

[47] M. Ghanes, J. P. Barbot, L. Fridman, and A. Levant, "A second order sliding mode differentiator with a variable exponent," in *2017 American Control Conference (ACC), Seattle, USA*, pp. 3300–3305, 2017.

[48] M. Ghanes, J. P. Barbot, L. Fridman, A. Levant, and R. Boisliveau, "A new varying gain exponent based differentiator/observer: an efficient balance between linear and sliding-mode algorithms," *IEEE Transactions on Automatic Control*, pp. 1–1, 2020.

[49] M. Mboup and S. Riachy, "Frequency-domain analysis and tuning of the algebraic differentiators," *International Journal of Control*, vol. 91, no. 9, pp. 2073–2081, 2018.

[50] W. Perruquetti and T. Floquet, "Homogeneous finite time observer for nonlinear systems with linearizable error dynamics," in *2007 46th IEEE Conference on Decision and Control*, pp. 390–395, 2007.

[51] A. Chalanga, S. Kamal, L. M. Fridman, B. Bandyopadhyay, and J. A. Moreno, "Implementation of super-twisting control: Super-twisting and higher order sliding-mode observer-based approaches," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 6, pp. 3677–3685, 2016.

[52] M. Ruderman, L. Fridman, and P. Pasolli, "Virtual sensing of load forces in hydraulic actuators using second- and higher-order sliding modes," *Control Engineering Practice*, vol. 92, p. 104151, 2019.

[53] O. Huber, V. Acary, and B. Brogliato, "Lyapunov stability and performance analysis of the implicit discrete sliding mode control," *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 3016–3030, 2016.

[54] O. Huber, V. Acary, B. Brogliato, and F. Plestan, "Implicit discrete-time twisting controller without numerical chattering: Analysis and experimental results," *Control Engineering Practice*, vol. 46, pp. 129–141, 2016.

[55] V. Acary, B. Brogliato, and Y. V. Orlov, "Chattering-free digital sliding-mode control with state observer and disturbance rejection," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1087–1101, 2012.

[56] O. Huber, V. Acary, and B. Brogliato, "Lyapunov stability analysis of the implicit discrete-time twisting control algorithm," *IEEE Transactions on Automatic Control*, vol. 65, no. 6, pp. 2619–2626, 2020.

[57] B. Wang, B. Brogliato, V. Acary, A. Boubakir, and F. Plestan, "Experimental comparisons between implicit and explicit implementations of discrete-time sliding mode controllers: Toward input and output chattering suppression," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 2071–2075, 2015.

[58] B. Brogliato, A. Polyakov, and D. Efimov, "The implicit discretization of the supertwisting sliding-mode control algorithm," *IEEE Transactions on Automatic Control*, vol. 65, no. 8, pp. 3707–3713, 2020.

[59] X. Xiong, R. Kikuuwe, and M. Yamamoto, "Backward-Euler discretization of second-order sliding mode control and super-twisting observer for accurate position control," in *ASME 2013 Dynamic Systems and Control Conference, Palo Alto, USA*, pp. 1–8, 2013.

[60] O. Huber, B. Brogliato, V. Acary, A. Boubakir, F. Plestan, and B. Wang, "Experimental results on implicit and explicit time-discretization of equivalent control-based sliding mode control," in *Recent Trends in Sliding Mode Control* (L. Fridman, J. Barbot, and F. Plestan, eds.), pp. 207–235, Institution of Engineering and Technology, 2016.

[61] B. Brogliato and A. Polyakov, "Digital implementation of sliding-mode control via the implicit method: A tutorial," *Int. J. Robust and Nonlinear Control*, Special issue.

[62] O. Huber and H. B. Oza, "Implicit numerical integration for the simulation and control of a non-smooth system with resets," in *2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, USA*, pp. 6551–6556, 2016.

[63] D. Efimov, A. Polyakov, A. Levant, and W. Perruquetti, "Realization and discretization of asymptotically stable homogeneous systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 11, pp. 5962–5969, 2017.

[64] J.-P. Barbot, A. Levant, M. Livne, and D. Lunz, "Discrete differentiators based on sliding modes," *Automatica*, vol. 112, p. 108633, 2020.

[65] A. Levant and M. Livne, "Exact differentiation of signals with unbounded higher derivatives," *IEEE Transactions on Automatic Control*, vol. 57, no. 4, pp. 1076–1080, 2012.

[66] J. Carvajal-Rubio, J. Sánchez-Torres, M. Defoort, and A. Loukianov, "On the discretization of robust exact filtering differentiators," *arXiv preprint arXiv:1911.09232*, 2019.

[67] J. E. Carvajal-Rubio, A. G. Loukianov, J. D. Sánchez-Torres, and M. Defoort, "On the discretization of a class of homogeneous differentiators," in *2019 16th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pp. 1–6, 2019.

[68] M. Avendaño, "Descartes' rule of signs is exact!," *Journal of Algebra*, vol. 324, no. 10, pp. 2884 – 2892, 2010.

[69] M. Wetzlinger, M. Reichhartinger, M. Horn, L. Fridman, and J. A. Moreno, "Semi-implicit discretization of the uniform robust exact differentiator," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 5995–6000, 2019.

[70] K. Alonzo, "A 3d state space formulation of a navigation kalman filter for autonomous vehicles," *Carnegie Mellon University. Technical Report CMU-RI-TR-94-19-REV*, vol. 2, p. 105, 1994.

[71] C. C. Y. Dorea, "Expected number of steps of a random optimization method," *Journal of Optimization Theory and Applications*, vol. 39, no. 2, pp. 165–171, 1983.

[72] Z. Galias and X. Yu, "Euler's discretization of single input sliding-mode control systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1726–1730, 2007.

[73] T. J. Ypma, "Historical development of the Newton-Raphson method," *SIAM Review*, vol. 37, no. 4, pp. 531–551, 1995.

[74] A. S. Householder, *The numerical treatment of a single nonlinear equation*. McGraw-Hill, 1970.

[75] L. Bairstow, *Applied aerodynamics*. Longmans, Green and Company, 1920.

[76] A. Polyakov, D. Efimov, and W. Perruquetti, "Finite-time and fixed-time stabilization: Implicit lyapunov function approach," *Automatica*, vol. 51, pp. 332–340, 2015.

[77] T. Sanchez, D. Efimov, A. Polyakov, J. A. Moreno, and W. Perruquetti, "A homogeneity property of discrete-time systems: Stability and convergence rates," *International Journal of Robust and Nonlinear Control*, vol. 29, no. 8, pp. 2406–2421, 2019.

[78] T. Sanchez, D. Efimov, A. Polyakov, and J. A. Moreno, "Homogeneous discrete-time approximation," *IFAC-PapersOnLine*, vol. 52, no. 16, pp. 19 – 24, 2019.